

AD-A168 361

GUIDE TO SIMULATION SCHEDULING APPENDICES B AND C(U)
CALIFORNIA UNIV BERKELEY OPERATIONS RESEARCH CENTER
R C LEACHMAN ET AL. JAN 86 ORC-86-1-APP-8/C

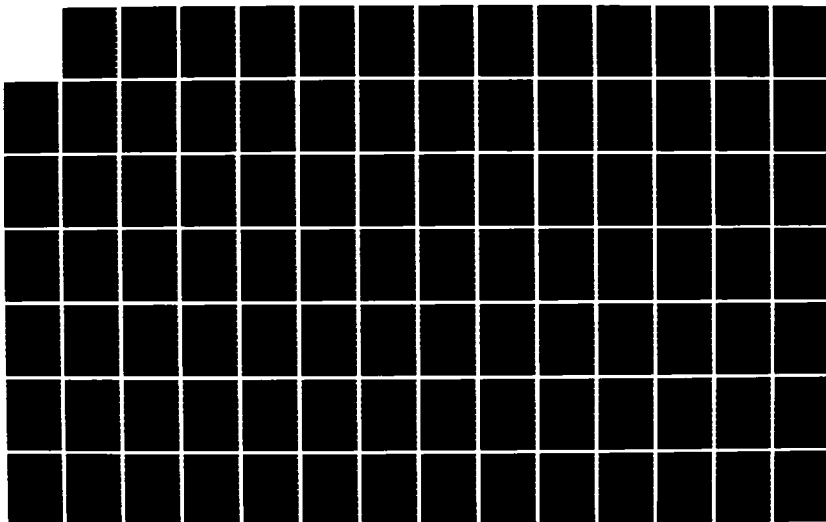
1/2

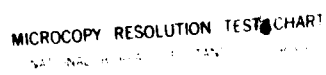
UNCLASSIFIED

N00014-76-C-0134

F/G 9/2

NL





AD-A168 361

OPERATIONS RESEARCH CENTER

APPENDICES B AND C OF
GUIDE TO SIMULATION SCHEDULING

by
Robert C. Leachman*
Sooyoung Kim*
Shirane Kyoung Chou

CM 86-1 Appendices B & C

January 1986

UNIVERSITY OF CALIFORNIA



BERKELEY

DTIC
ELECTE
JUN 3 1986
A

THIS DOCUMENT IS UNCLASSIFIED
DATE 10-15-86 BY 1045
ORIGINAL IS UNCLASSIFIED

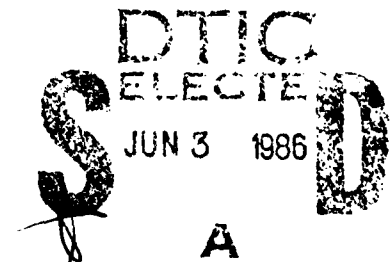
86 6 3 048

APPENDICES B AND C OF
GUIDE TO SIMULATION SCHEDULING

by
Robert C. Leachman*
Sooyoung Kim
Shrane Koung Chou

ORC 86-1 Appendices B & C

January 1986



*Operations Research Center, University of California, Berkeley, California.

This research was supported by the Office of Naval Research under Contract N00014-76-C-0134 with the University of California. Reproduction in whole or in part is permitted for any purpose of the United States Government.

This document has been approved
for public release and sale; its
distribution is unlimited.

TABLE OF CONTENTS

	Page
Appendix B. Listings of Execution Files	1
Appendix C. Listings of Fortran Programs	21

A1



APPENDIX B. Listings of Execution Files

1. TAPDSK SUBMIT
2. STARTDAY EXEC
3. CALENDAR EXEC
4. MILES EXEC
5. GEN EXEC
6. FTRAN EXEC
7. CAPDATE EXEC
8. SHOPCAP EXEC
9. REPDAT EXEC
10. SIMGEN EXEC
11. SIMSCHED EXEC
12. MICORG EXEC
13. MICORGR EXEC
14. BMICORG SUBMIT
15. BMICORGR SUBMIT
16. OUTREP EXEC
17. GRAPH EXEC

- 2 -

TAPDSK SUBMIT

TAPMOUNT SHIP • TAP1 NL DEN 1600

CMSDISK WRITEPW SHIPYARD

TAPTODSK SHIP DATA D (1 RECFM FB LRECL 60 BLOCK 1200

- 3 -

STARTDAY EXEC

EXEC USE FORTVS

FI 3 DISK START DATE A

LOAD STARTDAY(CLEAR START

FI • CLEAR

CALENDAR EXEC

EXEC USE FORTVS

FI 1 DISK FIRST YEAR A

FI 2 DISK WORKING DATES A

FI 4 DISK HOLIDAY DATE A

FI 8 DISK WORKING DAY A (RECFM F LRECL 132)

LOAD CALENDAR(CLEAR START

FI * CLEAR

MILES EXEC

EXEC USE FORTVS

FI 1 DISK SHIP DATA

FI 2 DISK BBMILES DATA

FI 3 DISK FIRST YEAR

FI 4 DISK WORKING DATES

FI 7 DISK START DATE

LOAD MILES(CLEAR START

FI * CLEAR

GEN EXEC

```
EXEC USE FORTVS
&TYPE **** ERASING OLD TEMPORARY FILES *****
ERASE BBS DATA A
ERASE BBSORT DATA A
ERASE BBALTO DATA A
* CHECK DISK SPACE
Q DISK A
&BEGTYPE

* BEFORE YOU START RUNNING THIS PROCEDURE, YOU SHOULD HAVE
  USED LESS THAN 80% OF DISK SPACE. CHECK QUOTA ABOVE AND
  ENTER Y IF WANT TO START OR N TO STOP NOW.
&END
&READ VARS &1
&IF .&1 NE .Y &GOTO -FIN
*
EXEC XSORT SHIP DATA A BBS DATA A 27 31
FI 1 DISK BBS DATA A
FI 2 DISK BBRES DATA A
LOAD RESLST (CLEAR START
ERASE BBS DATA A
FI * CLEAR

EXEC USE FORTVS
FI 1 DISK BBRES DATA
FI 2 DISK BBSHOPS DATA
FI 3 DISK SSHPLST DATA
FI 4 DISK BBNRES DATA
LOAD SHOPS(CLEAR START
FI * CLEAR

EXEC USE FORTVS
EXEC XSORT SHIP DATA A BBSORT DATA A 16 19 21 25 27 31
FI 10 DISK BBRES DATA A
FI 1 DISK BBSORT DATA A
FI 2 DISK BBALIST DATA A
FI 3 DISK BBWCLST DATA A
FI 4 DISK BBINTEN DATA A
FI 7 DISK BBANUM DATA A
LOAD INFILE1 (CLEAR START
ERASE BBSORT DATA A
FI * CLEAR

EXEC USE FORTVS
EXEC XSORT BBALIST DATA A BBALTO DATA A 12 16
FI 1 DISK BBALIST DATA A
FI 2 DISK BBALTO DATA A
FI 3 DISK BBFOLL DATA A
FI 4 DISK BBPRED DATA A
FI 7 DISK BBNET DATA A
LOAD FOLLPRED ( CLEAR START
ERASE BBALTO DATA A
FI * CLEAR
```

EXEC USE FORTVS
FI 1 DISK BBFOLL DATA A
FI 21 DISK BBPRED DATA A
FI 2 DISK BBNET DATA A
FI 8 DISK BBALIST DATA
FI 3 DISK BBRANK OUT A
FI 7 DISK BBRANK1 OUT A
EXEC USE FORTVS
LOAD RANKM (CLEAR START
-FIN
FI • CLEAR

FTRAN EXEC

EXEC USE FORTVS
&BEGTYPE

- IS THE FILE "BBTRC DATA" OF FLOW-TRANSFER INPUTS ALREADY COMPLETE?
IF YES, EXEC WILL COMPILE THE NETWORK.
Y IF YES, N IF YOU WANT TO ENTER NEW DATA FOR FLOW-TRANSFERS.

&END

&READ VARS &1

&IF .&1 EQ .Y &GOTO -CTRAN

•

FI 1 DISK BBTRC DATA A

LOAD FTRAN (CLEAR START

-CTRAN

FI 1 DISK BBTRC DATA A

FI 2 DISK BBFOLL DATA A

FI 3 DISK BBNET DATA A

FI 4 DISK BBALIST DATA A

FI 7 DISK BBTRAN DATA A

FI 12 DISK BBPRED DATA A

LOAD CTRAN (CLEAR START

FI • CLEAR

CAPDATE EXEC

EXEC USE FORTVS

FI 1 DISK FIRST YEAR A

FI 2 DISK WORKING DATES A

FI 3 DISK BBTCAP DATA A

FI 4 DISK START DATE A

FI 9 DISK SSPCAP DATE A

LOAD CAPDATE (CLEAR START

FI * CLEAR

SHOPCAP EXEC

EXEC USE FORTVS
&BEGTYPE

- DO YOU ALREADY HAVE THE FILE BBBCAP FOR SHOP CAPACITY INPUT ?
IF YES, EXEC WILL SKIP THE SHOPCAP AND CONTINUE TO PRECAP.
Y IF YES, N IF YOU WANT TO ENTER NEW DATA FOR SHOP CAPACITY.

&END

&READ VARS &1

&IF .&1 EQ .Y &GOTO -PRECAP

•

F1 1 DISK BBBCAP DATA A

LOAD SHOPCAP (CLEAR START

F1 • CLEAR

-PRECAP

EXEC USE FORTVS

F1 1 DISK BBNRES DATA

F1 2 DISK BBTCAP DATA

F1 3 DISK BBSHOPS DATA

F1 4 DISK BBBCAP DATA

F1 7 DISK BBRES DATA

F1 8 DISK BBVCAP DATA

F1 9 DISK BBNCAP DATA

F1 10 DISK SSPCAPO FILE

LOAD PRECAP (CLEAR START

F1 • CLEAR

REPDAT EXEC

EXEC USE FORTVS

FI 1 DISK FIRST YEAR A

FI 2 DISK WORKING DATES A

FI 3 DISK BBTREP DATA A

FI 4 DISK START DATE A

LOAD REPDAT (CLEAR START

FI * CLEAR

FI 1 DISK BBTCAP DATA A

FI 2 DISK BBTREP DATA A

FI 3 DISK BBSREP DATA A

LOAD REPCON1 (CLEAR START

FI * CLEAR

EXEC XSORT BBSREP DATA A BBBREP DATA A 1 4

FI 2 DISK START DATE A

FI 3 DISK FIRST YEAR A

FI 4 DISK WORKING DATES A

FI 7 DISK BBBREP DATA A

FI 8 DISK BBTREP DATA A

FI 9 DISK SSTREP DATA A

FI 11 DISK BBNREP DATA A

LOAD REPCON2 (CLEAR START

FI * CLEAR

SIMGEN EXEC

EXEC USE FORTVS

FI 2 DISK REWORK DATA A
FI 3 DISK BBALIST DATA A
FI 4 DISK BBTEST DATA A
FI 7 DISK BALISTR DATA A
FI 8 DISK BTRANKR DATA A
FI 9 DISK BDURR DATA A
FI 10 DISK BPROBR DATA A
FI 11 DISK BTFOLLR DATA A
FI 12 DISK BTATTR DATA A
FI 13 DISK BTRCR DATA A
LOAD COVERT (CLEAR START
FI * CLEAR

EXEC USE FORTVS

FI 1 DISK BALISTR DATA A
FI 2 DISK BBPRED DATA A
FI 3 DISK BBNET DATA A
FI 4 DISK BBFOLL DATA A
FI 7 DISK BNETR DATA A
FI 8 DISK BFOLR DATA A
FI 9 DISK BPREDR DATA A
FI 13 DISK BTRCR DATA A
LOAD RPFOL (CLEAR START
FI * CLEAR

EXEC USE FORTVS

FI 1 DISK BRANUM DATA A
FI 2 DISK BFOLR DATA A
FI 3 DISK BRWCLST DATA A
FI 4 DISK BBANUM DATA A
FI 7 DISK BBALIST DATA A
FI 8 DISK BNETR DATA A
FI 9 DISK BBWCLST DATA A
FI 10 DISK BBINTEN DATA A
FI 11 DISK BBTEST DATA A
FI 13 DISK BTRCR DATA A
FI 14 DISK BRINTEN DATA A
FI 17 DISK BALISTR DATA A
FI 20 DISK RRES DATA A
FI 21 DISK BBRES DATA A
LOAD RAB (CLEAR START
FI * CLEAR

SIMSCHEDE EXEC

&CONTROL OFF

• INTERACTIVE EXECS FOR RUNNING MICORG

&BEGTYPE

***** SIMSCHEDE *****

SELECT ONE OF THE FOLLOWING OPTION CODES :

- 1 : INTERACTIVE RUN WITHOUT TEST-REWORK LOOPS (DETERMINISTIC RUN)
- 2 : INTERACTIVE RUN WITH TEST-REWORK LOOPS (STOCHASTIC RUN)
- 3 : BATCH RUN WITHOUT TEST-REWORK LOOPS (DETERMINISTIC RUN)
- 4 : BATCH RUN WITH TEST-REWORK LOOPS (STOCHASTIC RUN)

- BATCH RUN WILL BE ACTUALLY SUBMITTED AFTER 22:00 (NIGHT OPTION USED)

&END

&READ VARS &1

&IF .&1 GT .2 &GOTO -BTCH

&BEGTYPE

&END

&IF .&1 EQ .1 EXEC MICORG

&IF .&1 EQ .2 EXEC MICORGR

&GOTO -FIN

-BTCH

&BEGTYPE

ENTER THE LIMIT OF CPU SECONDS FOR YOUR BATCH RUN (0 IF NONE)

&END

&READ VARS &2

&IF .&1 EQ .3 &GOTO -BT1

&IF .&1 EQ .4 &GOTO -BT2

-BT1

&IF .&2 EQ .0 &TYPE SUBMIT BMICORG CLASS B NIGHT

&IF .&2 NE .0 &TYPE SUBMIT BMICORG CLASS B SEC &2 NIGHT

&GOTO -FIN

-BT2

&IF .&2 EQ .0 &TYPE SUBMIT BMICORGR CLASS B NIGHT

&IF .&2 NE .0 &TYPE SUBMIT BMICORGR CLASS B SEC &2 NIGHT

-FIN

&EXIT

MICORG EXEC

EXEC USE FORTVS

FI 1 DISK BPARAMR DATA
FI 2 DISK BBFOLL DATA
FI 3 DISK BBPRED DATA
FI 4 DISK BBANUM DATA
FI 7 DISK BBALIST DATA
FI 8 DISK BBNET DATA
FI 9 DISK BBWCLST DATA
FI 10 DISK BBINTEN DATA
FI 11 DISK BBRANK OUT
FI 12 DISK BBTRAN DATA
FI 13 DISK BBSHOPS DATA
FI 14 DISK BBVCAP DATA
FI 15 DISK BBTCAP DATA
FI 16 DISK BBTREP DATA
FI 17 DISK BBRAND DATA
FI 33 DISK BBNRES DATA
FI 34 DISK BBNCAP DATA
FI 35 DISK BBNREP DATA

*** OUTPUT FILES:**

FI 25 DISK BBACWC OUT
FI 26 DISK BBACSHP OUT
FI 27 DISK BBTMILE OUT
FI 57 DISK BBCMILE OUT
FI 28 DISK BBTMR OUT
FI 36 DISK SCHEDULE OUT (RECFM F LRECL 130)
FI 37 DISK BBUPDATE OUT
FI 40 DISK STATUS OUT

***INPUT FOR CALENDAR CONVERSION**

FI 52 DISK START DATE
FI 53 DISK FIRST YEAR
FI 54 DISK WORKING DATES
LOAD MICORG (CLEAR START

MICORGR EXEC

EXEC USE FORTVS

FI 1 DISK BPARAMR DATA

FI 2 DISK BFOLR DATA

FI 3 DISK BPREDR DATA

FI 4 DISK BRANUM DATA

FI 7 DISK BALISTR DATA

FI 8 DISK BNETR DATA

FI 9 DISK BRWCLST DATA

FI 10 DISK BRINTEN DATA

FI 11 DISK BBRANK OUT

FI 12 DISK BBTRAN DATA

FI 13 DISK BBSHOPS DATA

FI 14 DISK BBVCAP DATA

FI 15 DISK BBTCAP DATA

FI 16 DISK BBTREP DATA

FI 17 DISK BBRAND DATA

FI 18 DISK BTFOLLR DATA

FI 19 DISK BTATTR DATA

FI 20 DISK BTRANKR DATA

FI 30 DISK BDURR DATA

FI 31 DISK BPROBR DATA

FI 33 DISK BBNRES DATA

FI 34 DISK BBNCAP DATA

FI 35 DISK BBNREP DATA

* OUTPUT FILES:

FI 25 DISK BBACWC OUT

FI 26 DISK BBACSHP OUT

FI 27 DISK BBTMILE OUT

FI 57 DISK BBCMILE OUT

FI 28 DISK BBTMHR OUT

FI 36 DISK SCHEDULE OUT (RECFM F LRECL 130

FI 37 DISK BRUPDATE OUT

FI 40 DISK STATUS OUT

*INPUT FOR CALENDAR CONVERSION

FI 52 DISK START DATE

FI 53 DISK FIRST YEAR

FI 54 DISK WORKING DATES

LOAD MICORG (CLEAR START

FI * CLEAR

BMICORG SUBMIT

SET CMSTYPE HT

CMSDISK READPW DYNPROD4

CMSDISK WRITEPW DYNPROD4

EXEC USE FORTVS

FI 1 DISK BPARAMR DATA D

FI 2 DISK BBFOLL DATA D

FI 3 DISK BBPRED DATA D

FI 4 DISK BBANUM DATA D

FI 7 DISK BBALIST DATA D

FI 8 DISK BBNET DATA D

FI 9 DISK BBWCLST DATA D

FI 10 DISK BBINTEN DATA D

FI 11 DISK BBRANK OUT D

FI 12 DISK BBTRAN DATA D

FI 13 DISK BBSHOPS DATA D

FI 14 DISK BBVCAP DATA D

FI 15 DISK BBTCAP DATA D

FI 16 DISK BBTREP DATA D

FI 17 DISK BBRAND DATA D

FI 33 DISK BBNRES DATA D

FI 34 DISK BBNCAP DATA D

FI 35 DISK BBNREP DATA D

* OUTPUT FILES:

FI 25 DISK BBACWC OUT D (PERM

FI 26 DISK BBACSHP OUT D (PERM

FI 27 DISK BBTMILE OUT D (PERM

FI 57 DISK BBCMILE OUT D (PERM

FI 28 DISK BBTMHR OUT D (PERM

FI 36 DISK SCHEDULE OUT D (PERM RECFM F LRECL 130

FI 37 DISK BBUPDATE OUT D (PERM

FI 40 DISK STATUS OUT D (PERM

*INPUT FOR CALENDAR CONVERSION

FI 52 DISK START DATE D

FI 53 DISK FIRST YEAR D

FI 54 DISK WORKING DATES D

LOAD BMICORG (CLEAR START

BMICORGR SUBMIT

SET CMSTYPE HT

CMSDISK READPW DYNPROD4

CMSDISK WRITEPW DYNPROD4

EXEC USE FORTVS

FI 1 DISK BPARAMR DATA D

FI 2 DISK BFOLR DATA D

FI 3 DISK BPREDR DATA D

FI 4 DISK BRANUM DATA D

FI 7 DISK BALISTR DATA D

FI 8 DISK BNETR DATA D

FI 9 DISK BRWCLST DATA D

FI 10 DISK BRINTEN DATA D

FI 11 DISK BBRANK OUT D

FI 12 DISK BBTRAN DATA D

FI 13 DISK BBSHOPS DATA D

FI 14 DISK BBVCAP DATA D

FI 15 DISK BBTCAP DATA D

FI 16 DISK BBTREP DATA D

FI 17 DISK BBRAND DATA D

FI 18 DISK BTFOLLR DATA D

FI 19 DISK BTATTR DATA D

FI 20 DISK BTRANKR DATA D

FI 30 DISK BDURR DATA D

FI 31 DISK BPROBR DATA D

FI 33 DISK BBNRES DATA D

FI 34 DISK BBNCAP DATA D

FI 35 DISK BBNREP DATA D

*** OUTPUT FILES:**

FI 25 DISK BBACWC OUT D (PERM

FI 26 DISK BBACSHP OUT D (PERM

FI 27 DISK BBTMILE OUT D (PERM

FI 57 DISK BBCMILE OUT D (PERM

FI 28 DISK BBTMHR OUT D (PERM

FI 36 DISK SCHEDULE OUT D (PERM RECFM F LRECL 130

FI 37 DISK BRUPDATE OUT D (PERM

FI 40 DISK STATUS OUT D (PERM

***INPUT FOR CALENDAR CONVERSION**

FI 52 DISK START DATE D

FI 53 DISK FIRST YEAR D

FI 54 DISK WORKING DATES D

LOAD BMICORG (CLEAR START

FI * CLEAR

OUTREP EXEC

EXEC USE FORTVS

ROUTE PRT ETCH1

* INTERACTIVE EXECS FOR RUNNING OUTPUT REPORT

&BEGTYPE

SELECT ONE OF THE FOLLOWING OPTION CODES FOR OUTPUT (REPORT OUT):

0 : PRINT

1 : WRITE ON DISK WITH NAME 'REPORT OUT'

&END

&READ VARS &1

&IF .&1 EQ .0 FI 10 PRINT (RECFM FBA LRECL 130 BLOCK 130

&IF .&1 EQ .1 FI 10 DISK REPORT OUT (RECFM FBA LRECL 130 BLOCK 130

FI 1 DISK START DATE

FI 2 DISK BBTREP DATA

FI 32 DISK SSTREP DATA

FI 3 DISK BBRES DATA

FI 4 DISK SSHPLST DATA

FI 7 DISK SSMILES DATA

FI 8 DISK STATUS OUT

FI 25 DISK BBACWC OUT

FI 26 DISK BBACSHP OUT

FI 27 DISK BBTMILE OUT

FI 57 DISK BBCMILE OUT

FI 28 DISK BBTMHR OUT

FI 11 DISK SSPWC0 FILE(RECFM F LRECL 100

FI 12 DISK SSPSHOP0 FILE(RECFM F LRECL 100

FI 13 DISK SSPMILE0 FILE(RECFM F LRECL 100

FI 14 DISK TOTAL FILE(RECFM F LRECL 100

FI 15 DISK SSPWC1 FILE(RECFM F LRECL 100

FI 16 DISK SSPSHOP1 FILE(RECFM F LRECL 100

FI 17 DISK SSPMILE1 FILE(RECFM F LRECL 100

LOAD SSP (CLEAR START

FI * CLEAR

GRAPH EXEC

&TRACE OFF
SET BLIP OFF
EXEC USE DISSPLA
LOAD GRAPH (NOAUTO CLEAR NOMAP
&IF &RC NE 0 &EXIT &RC
FILEDEF 6 TERM
FILEDEF 5 TERM
FI 1 DISK SSPMILE0 FILE
FI 2 DISK SSPMILE1 FILE
FI 3 DISK SSPWC0 FILE
FI 4 DISK SSPWC1 FILE
FI 7 DISK SSPSHOP0 FILE
FI 8 DISK SSPSHOP1 FILE
FI 9 DISK TOTAL FILE
FI 10 DISK SSPCAPO FILE
FI 12 DISK STATUS OUT
EXEC GRSAVE
START
EXEC GRRESET
SET BLIP ON
&EXIT &RC

KEEP EXEC

&CONTROL OFF

• INTERACTIVE EXECS FOR SAVING OUTPUTS USING PROJECT NAME

&BEGTYPE

***** KEEP *****

THIS EXEC RENAMES THE OUTPUTS WITH THE PROJECT I.D. TO KEEP THEM

ENTER THE PROJECT I.D. (MAX. 8 CHARACTERS)

&END

&READ VARS &1

• RENAME INPUT AND INTERMEDIATE FILES

RENAME SHIP DATA A &1 DATA A

RENAME FIRST YEAR A &1 FIRST A

RENAME WORKING DATES A &1 DATES A

RENAME START DATE A &1 START A

RENAME BBTRC DATA A &1 BBTRC A

RENAME BBTCAP DATA A &1 BBTCAP A

RENAME BBBCAP DATA A &1 BBBCAP A

RENAME SSPCAP DATA A &1 SSPCAP A

RENAME BBVCAP DATA A &1 BBVCAP A

RENAME BBNCAP DATA A &1 BBNCAP A

RENAME BBTREP DATA A &1 BBTREP A

RENAME BBNREP DATA A &1 BBNREP A

RENAME SSTREP DATA A &1 SSTREP A

RENAME REWORK DATA A &1 REWORK A

RENAME RRES DATA A &1 RRES A

RENAME BBRAND DATA A &1 BBRAND A

• OUTPUTS

RENAME SCHEDULE OUT A &1 SCHEDULE A

RENAME REPORT OUT A &1 REPORT A

RENAME STATUS OUT A &1 STATUS A

&EXIT

RETRIEVE EXEC

&CONTROL OFF

* INTERACTIVE EXECS FOR RETRIEVING FILES TO ORIGINAL NAMES

&BEGTYPE

***** RETRIEVE *****

THIS EXEC RETRIEVES THE FILES TO THEIR ORIGINAL NAMES AND MODES.

ENTER THE PROJECT I.D. (MAX. 8 CHARACTERS)

&END

&READ VARS &1

* RENAME INPUT AND INTERMEDIATE FILES

RENAME &1 DATA A SHIP DATA A

RENAME &1 FIRST A FIRST YEAR A

RENAME &1 DATES A WORKING DATES A

RENAME &1 START A START DATE A

RENAME &1 BBTRC A BBTRC DATA A

RENAME &1 BBTCAP A BBTCAP DATA A

RENAME &1 BBBCAP A BBBCAP DATA A

RENAME &1 SSPCAP A SSPCAP DATA A

RENAME &1 BBVCAP A BBVCAP DATA A

RENAME &1 BBNCAP A BBNCAP DATA A

RENAME &1 BBTREP A BBTREP DATA A

RENAME &1 BBNREP A BBNREP DATA A

RENAME &1 SSTREP A SSTREP DATA A

RENAME &1 REWORK A REWORK DATA A

RENAME &1 RRES A RRES DATA A

RENAME &1 BBRAND A BBRAND DATA A

* OUTPUTS

RENAME &1 SCHEDULE A SCHEDULE OUT A

RENAME &1 REPORT A REPORT OUT A

RENAME &1 STATUS A STATUS OUT A

&EXIT

APPENDIX C. Listings of Fortran Programs

1. STARTDAY
2. CALENDAR
3. MILES
4. RESLST
5. SHOPS
6. INFILE1
7. FOLLPRED
8. RANKM
9. FTRAN
10. CTRAN
11. CAPDATE
12. SHOPCAP
13. PRECAP
14. REPDAT
15. REPCON1
16. REPCON2
17. COVERT
18. RPFOL
19. RAB
20. MICORG
21. BMICORG
22. SSP
23. GRAPH

```

C *****STAC0010
C                                     STAC0020
C          STARTDAY  FORTRAN          STAC0030
C          -----                     STAC0040
C          THIS PROGRAM GETS THE STARTING DATE OF THE CURRENT STAC0050
C          PROJECT USED AS A BASIS FOR EXCHANGING WORKING DAY STAC0060
C          AND CALENDAR DAY IN INPUT AND OUTPUT FORMAT STAC0070
C                                     STAC0080
C *****STAC0090
C          INTEGER SMONTH,SDAY,SYEAR STAC0100
C          10 WRITE(6,101) STAC0110
C          101 FORMAT(//,3X,'INPUT THE STARTING DATE OF THE CURRENT PROJECT?',/, STAC0120
C             +3X,'MONTH, DAY, YEAR') STAC0130
C             READ(5,*)SMONTH,SDAY,SYEAR STAC0140
C             WRITE(6,103)SMONTH,SDAY,SYEAR STAC0150
C          103 FORMAT(//,3X,'YOU HAVE INPUT DATE:',2I5,1X,15,2X,'AS THE STARTING STAC0160
C             +DATE OF THE CURRENT',/,3X,'PROJECT',//,3X,'IT IS CORRECT OR NOT? STAC0170
C             +IF YES PLEASE INPUT 1, IF NOT PLEASE INPUT 0') STAC0180
C             READ(5,*)INDEX STAC0190
C             IF(INDEX.EQ.0) GO TO 10 STAC0200
C             WRITE(3,102)SMONTH,SDAY,SYEAR STAC0210
C          102 FORMAT(10X,3I5) STAC0220
C             STOP STAC0230
C             END STAC0240

```

```

FILE: CALENDAR FORTRAN A VM/SP CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY
C *****
C
C          CALENDAR FORTRAN
C          -----
C *      THIS PROGRAM GENERATE THE WORKING DATES TABLE *
C *      THE RESULT PUT INTO THE FILE "WORKING DATES" *
C *      THE FILE INCLUDES 10 YEARS DATA FROM THE BEGINNING *
C *      OF THE FIRST YEAR WHICH YOU WILL BE ASKED TO INPUT, IN THE *
C *      WORKING DATES TABLE "1" MEANS WORKING DAY, "0" HOLIDAY *
C *
C *****
C      INTEGER HOLIDAY(12,31,10), HOLIDAY2(100,3), DATE(31), MON(12)
C      INTEGER SAT, SUN, MONTH, DAY, YEAR, YEARN, DURATN, PVM, SUNA
C      INTEGER SYEAR, SMON, SYF, IMONTH(31), LYEAR
C CHOOSING THE WORKING DAY POLICY
C      WRITE(6,101)
101  FORMAT(//,3X,'IN YOUR FACILITY, IS SATURDAY A WORKING DAY?',//,
+3X,'IF YES PLEASE INPUT 1, IF NOT PLEASE INPUT 0')
C      READ(5,*)INDEX
C      WRITE(6,119)
119  FORMAT(//,3X,'IN YOUR FACILITY, IS SUNDAY A WORKING DAY?',//,
+3X,'IF YES PLEASE INPUT 1, IF NOT PLEASE INPUT 0')
C      READ(5,*)INDEXS
C      IF(INDEXS.EQ.1)GOTO 150
C      IF(INDEX.EQ.0)GOTO 106
C      WRITE(6,107)
107  FORMAT(//,3X,'OH! YOU STILL WORK ON SATURDAY.')
```

GOTO 106
150 IF(INDEX.EQ.0)GOTO 151
WRITE(6,152)
152 FORMAT(//,3X,'NO REST IN WHOLE WEEK? BE CAREFUL YOUR HEALTH !')
GOTO 106
151 WRITE(6,153)
153 FORMAT(//,3X,'OH! YOU WORK ON SUNDAY AND REST ON SATURDAY !')

C ASK INPUT ANY SUNDAY WITHIN THE DURATION, THE PROGRAM WILL DELETE THE
C NORMAL HOLIDAY IN THE WORKING DAY TABLE.
106 WRITE(6,154)
154 FORMAT(//,3X,'ENTER THE YEAR WHICH YOU WANT AS THE STARTING YEAR'
+/,3X,'THE PROGRAM WILL GENERATE THE WORKING DAY TABLE FOR 10 YEAR'
+S USE',/,3X,'STARTING FROM THE YEAR YOU HAVE INPUT')
READ(5,*)SYEAR
EYEAR=SYEAR+9
IF(INDEXS.EQ.0)GOTO 630
IF(INDEX.EQ.1)GOTO 459
630 WRITE(6,102)SYEAR,EYEAR
102 FORMAT(//,3X,'ENTER ANY DATE OF A SUNDAY BETWEEN YEAR',16,3X,'AND
+YEAR',16,3X,/,3X,'(MM,DD,YY)')
READ(5,*)MONTH,DAY,LYEAR
YEAR=1900+LYEAR
IF((YEAR.GE.SYEAR).AND.(YEAR.LE.EYEAR)) GOTO 620
WRITE(6,615)
615 FORMAT(//,3X,'THE DATE YOU JUST INPUT IS OUT OF THE RANGE WHICH YOU
+J ASSIGNED',/,3X,'PLEASE REVISE IT!')
GOTO 630
620 WRITE(6,103)MONTH,DAY,YEAR,SYEAR,EYEAR
103 FORMAT(//,3X,'YOU HAVE INPUT DATE',214,16,'. IT IS A SUNDAY'
+/,3X,'BETWEEN YEAR',16,3X,'AND YEAR',16)

DURATN=DAY	CAL00560
IF(MONTH.EQ.1) GOTO 450	CAL00570
PVM=MONTH-1	CAL00580
DO 400 I=PVM,1,-1	CAL00590
IF((I.EQ.2).OR.(I.EQ.4).OR.(I.EQ.6).OR.(I.EQ.9).OR.(I.EQ.11))GOTO	CAL00600
+410	CAL00610
DURATN=DURATN+31	CAL00620
GOTO 400	CAL00630
410 IF(I.EQ.2) GOTO 420	CAL00640
DURATN=DURATN+30	CAL00650
GO TO 400	CAL00660
420 LL=MOD(YEAR,4)	CAL00670
IF(LL.EQ.0)GO TO 430	CAL00680
DURATN=DURATN+28	CAL00690
GO TO 400	CAL00700
430 DURATN=DURATN+29	CAL00710
400 CONTINUE	CAL00720
C WRITE(6,*)DURATN	CAL00730
450 NYEAR=YEAR	CAL00740
472 IF(NYEAR.EQ.SYEAR)GOTO 470	CAL00750
NYLAR=NYEAR-1	CAL00760
MYEAR=MOD(NYLAR,4)	CAL00770
IF(MYEAR.EQ.0)GOTO 471	CAL00780
DURATN=DURATN+365	CAL00790
GOTO 472	CAL00800
471 DURATN=DURATN+366	CAL00810
GOTO 472	CAL00820
470 SUNA=MOD(DURATN,7)	CAL00830
C WRITE(6,*)DURATN	CAL00840
IF(SUNA.GT.0) GO TO 455	CAL00850
SUN=SYNA+7	CAL00860
SAT=6	CAL00870
GOTO 459	CAL00880
455 IF(SUNA.EQ.1)GOTO 458	CAL00890
SAT=SYNA-1	CAL00900
SUN=SYNA	CAL00910
GOTO 459	CAL00920
458 SUN=SYNA	CAL00930
SAT=SYNA+6	CAL00940
C GENERATE WORKING DAY TABLE, SET ALL DATE AS WORKING DAY.	CAL00950
459 DO 700 I=1,12	CAL00960
DO 701 J=1,31	CAL00970
DO 702 K=1,10	CAL00980
HOLIDA(I,J,K)=1	CAL00990
702 CONTINUE	CAL01000
701 CONTINUE	CAL01010
700 CONTINUE	CAL01020
DO 480 K=1,10	CAL01030
DO 485 I=1,12	CAL01040
IF((I.EQ.1).OR.(I.EQ.2).OR.(I.EQ.5).OR.(I.EQ.7))GOTO 485	CAL01050
IF((I.EQ.8).OR.(I.EQ.10).OR.(I.EQ.12))GOTO 485	CAL01060
HOLIDA(I,31,K)=0	CAL01070
IF(I.EQ.2)GOTO 465	CAL01080
IY=K+SYEAR-1	CAL01090
KY=MOD(IY,4)	CAL01100

HOLICA(I,30,K)=0	CAL01110
IF(KY.EQ.0)GOTO 485	CAL01120
HOLICA(I,29,K)=0	CAL01130
485 CONTINUE	CAL01140
480 CONTINUE	CAL01150
C CALCULATE THE NORMAL WORKING DAY	CAL01160
IF(INDEXS.EQ.1)GOTO 475	CAL01170
IF(INDEX.EQ.C)GOTO 46C	CAL01180
C ELIMINATE ALL THE SUNDAYS FROM THE WORKING DAY TABLE.	CAL01190
DO 800 K=1,10	CAL01200
DO 801 I=1,12	CAL01210
IF((I.EQ.2).OR.(I.EQ.4).OR.(I.EQ.6).OR.(I.EQ.9).OR.(I.EQ.11)) GOTO	CAL01220
+810	CAL01230
820 HOLICA(I,SUN,K)=0	CAL01240
SUN=SUN+7	CAL01250
IF(SUN.LE.31)GOTO 820	CAL01260
SUN=SUN-31	CAL01270
GOTO 801	CAL01280
810 IF(I.EQ.2)GOTO 830	CAL01290
815 HOLICA(I,SUN,K)=0	CAL01300
SUN=SUN+7	CAL01310
IF(SUN.LE.30)GOTO 815	CAL01320
SUN=SUN-30	CAL01330
GOTO 801	CAL01340
830 JM=K+SYEAR-1	CAL01350
KM=MOD(JM,4)	CAL01360
IF(KM.EQ.0)GOTO 840	CAL01370
835 HOLICA(I,SUN,K)=0	CAL01380
SUN=SUN+7	CAL01390
IF(SUN.LE.28)GOTO 835	CAL01400
SUN=SUN-28	CAL01410
GOTO 801	CAL01420
840 HOLICA(I,SUN,K)=0	CAL01430
SUN=SUN+7	CAL01440
IF(SUN.LE.29)GOTO 840	CAL01450
SUN=SUN-29	CAL01460
801 CONTINUE	CAL01470
800 CONTINUE	CAL01480
WRITE(6,850)	CAL01490
850 FORMAT(//,3X,'INPUT ALL HOLIDAYS (OTHER THAN SUNDAYS): MM,DD,YY',/	CAL01500
+/,3X,'END OF DATA PLEASE PRESS RETURN TWICE',/,3X,'IF THE HOLIDAY	CAL01510
+ATE IS THE SAME EVERY YEAR, THEN INPUT : MM,DD,0')	CAL01520
GOTO 305	CAL01530
C ELIMINATE ALL THE SUNDAYS & SATURDAYS FROM THE WORKING DAY TABLE	CAL01540
460 DO 210 K=1,10	CAL01550
DO 220 I=1,12	CAL01560
IF((I.EQ.2).OR.(I.EQ.4).OR.(I.EQ.6).OR.(I.EQ.9).OR.(I.EQ.11)) GOTO	CAL01570
+120	CAL01580
111 HOLICA(I,SAT,K)=0	CAL01590
HOLICA(I,SUN,K)=0	CAL01600
SAT=SUN+6	CAL01610
SUN=SAT+1	CAL01620
IF(SAT.GE.32) GOTO 112	CAL01630
IF(SUN.LE.31) GOTO 111	CAL01640
HOLICA(I,SAT,K)=0	CAL01650

	SUN=SUN-31	CAL0166C
	SAT=SUN+6	CAL01670
	GOTO 220	CAL01680
112	SUN=SUN-31	CAL01690
	SAT=SAT-31	CAL01700
	GOTO 220	CAL01710
120	IF(1.EQ.2) GOTO 130	CAL01720
121	HOLIDAY(I,SAT,K)=0	CAL01730
	HOLIDAY(I,SUN,K)=0	CAL01740
	SAT=SUN+6	CAL01750
	SUN=SAT+1	CAL01760
	IF(SAT.GE.31) GOTO 122	CAL01770
	IF(SUN.LE.30) GOTO 121	CAL01780
	HOLIDAY(I,SAT,K)=0	CAL01790
	HOLIDAY(I,31,K)=0	CAL01800
	SUN=SUN-30	CAL01810
	SAT=SUN+6	CAL01820
	GOTO 220	CAL01830
112	SUN=SUN-30	CAL01840
	SAT=SAT-30	CAL01850
	HOLIDAY(I,31,K)=0	CAL01860
	GOTO 220	CAL01870
130	JK=K+YEAR-1	CAL01880
	KL=MOD(JK,4)	CAL01890
	IF(KL.EQ.0) GOTO 140	CAL01900
131	HOLIDAY(I,SAT,K)=0	CAL01910
	HOLIDAY(I,SUN,K)=0	CAL01920
	SAT=SUN+6	CAL01930
	SUN=SAT+1	CAL01940
	IF(SAT.GE.29) GOTO 132	CAL01950
	IF(SUN.LE.28) GOTO 131	CAL01960
	HOLIDAY(I,SAT,K)=0	CAL01970
	HOLIDAY(I,29,K)=0	CAL01980
	HOLIDAY(I,30,K)=0	CAL01990
	HOLIDAY(I,31,K)=0	CAL02000
	SUN=SUN-28	CAL02010
	SAT=SUN+6	CAL02020
	GOTO 220	CAL02030
132	SUN=SUN-26	CAL02040
	SAT=SAT-26	CAL02050
	HOLIDAY(I,29,K)=0	CAL02060
	HOLIDAY(I,30,K)=0	CAL02070
	HOLIDAY(I,31,K)=0	CAL02080
	GOTO 220	CAL02090
140	HOLIDAY(I,SAT,K)=0	CAL02100
	HOLIDAY(I,SUN,K)=0	CAL02110
	SAT=SUN+6	CAL02120
	SUN=SAT+1	CAL02130
	IF(SAT.GE.30) GOTO 141	CAL02140
	IF(SUN.LE.29) GOTO 140	CAL02150
	HOLIDAY(I,SAT,K)=0	CAL02160
	HOLIDAY(I,30,K)=0	CAL02170
	HOLIDAY(I,31,K)=0	CAL02180
	SUN=SUN-29	CAL02190
	SAT=SUN+6	CAL02200

GOTO 220	CAL02210
142 SUN=SUN-29	CAL02220
SAT=SAT-29	CAL02230
HOLIDAY(I,30,K)=0	CAL02240
HOLIDAY(I,31,K)=0	CAL02250
GOTO 220	CAL02260
220 CONTINUE	CAL02270
210 CONTINUE	CAL02280
WRITE(6,301)	CAL02290
301 FORMAT(//,3X,'INPUT ALL HOLIDAYS (OTHER THAN SATURDAYS & SUNDAYS):	CAL02300
+MM,DD,YY',/,3X,'END OF DATA PRESS RETURN TWICE',/,3X,	CAL02310
+ 'IF THE HOLIDAY DATE IS THE SAME EVERY YEAR,/,3X,'THEN INPUT :	CAL02320
+MM,DD,0')	CAL02330
GOTO 305	CAL02340
C ELIMINATE ALL THE SATURDAYS FROM THE WORKING DAY TABLE.	CAL02350
475 IF(INDEX.EQ.1)GOTO 476	CAL02360
DO 900 K=1,10	CAL02370
DO 901 I=1,12	CAL02380
IF((I.EQ.2).OR.(I.EQ.4).OR.(I.EQ.6).OR.(I.EQ.9).OR.(I.EQ.11)) GOTO	CAL02390
+910	CAL02400
920 HOLIDAY(I,SAT,K)=0	CAL02410
SAT=SAT+7	CAL02420
IF(SAT.LE.31)GOTO 920	CAL02430
SAT=SAT-31	CAL02440
GOTO 901	CAL02450
910 IF(I.EQ.2)GOTO 930	CAL02460
915 HOLIDAY(I,SAT,K)=0	CAL02470
SAT=SAT+7	CAL02480
IF(SAT.LE.30)GOTO 915	CAL02490
SAT=SAT-30	CAL02500
GOTO 901	CAL02510
930 JM=K+SYEAR-1	CAL02520
KM=MOD(JM,4)	CAL02530
IF(KM.EQ.0)GOTO 940	CAL02540
935 HOLIDAY(I,SAT,K)=0	CAL02550
SAT=SAT+7	CAL02560
IF(SAT.LE.28)GOTO 935	CAL02570
SAT=SAT-28	CAL02580
GOTO 901	CAL02590
940 HOLIDAY(I,SAT,K)=0	CAL02600
SAT=SAT+7	CAL02610
IF(SAT.LE.29)GOTO 940	CAL02620
SAT=SAT-29	CAL02630
901 CONTINUE	CAL02640
900 CONTINUE	CAL02650
WRITE(6,950)	CAL02660
950 FORMAT(//,3X,'INPUT ALL HOLIDAYS (OTHER THAN SATURDAYS): MM,DD,YY'	CAL02670
+/,3X,'END OF DATA PLEASE PRESS RETURN TWICE',/,3X,'IF THE HOLIDAY	CAL02680
+ DATE IS THE SAME DAY EVERY YEAR, THEN INPUT : MM,DD,0')	CAL02690
GOTO 305	CAL02700
476 WRITE(6,951)	CAL02710
951 FORMAT(//,3X,'INPUT ALL HOLIDAYS : MM,DD,YY',/	CAL02720
+/,3X,'END OF DATA PLEASE PRESS RETURN TWICE',/,3X,'IF THE HOLIDAY	CAL02730
+ DATE IS THE SAME EVERY YEAR, THEN INPUT : MM,DD,0')	CAL02740
305 DO 210 I=1,100	CAL02750

READ(5,*,END=320)111,JJJ,KKK	CAL02760
HOLID2(1,1)=111	CAL02770
HOLID2(1,2)=JJJ	CAL02780
IF(KKK.EQ.0)GOTO 388	CAL02790
HOLID2(1,3)=KKK+1900	CAL02800
GOTO 310	CAL02810
388 HOLID2(1,3)=KKK	CAL02820
310 CONTINUE	CAL02830
320 DO 330 I=1,100	CAL02840
I1=HOLID2(1,1)	CAL02850
JJ=HOLID2(1,2)	CAL02860
K1=HOLID2(1,3)	CAL02870
C WRITE(6,*)I1,JJ,K1,1	CAL02880
IF(I1.EQ.0)GOTO 340	CAL02890
IF(K1.EQ.0)GOTO 321	CAL02900
KK=HOLID2(1,3)-SYEAR+1	CAL02910
HOLIDA(I1,JJ,KK)=C	CAL02920
GO TO 330	CAL02930
321 DO 322 K=1,10	CAL02940
HOLIDA(I1,JJ,K)=C	CAL02950
C WRITE(6,*)I1,JJ,K	CAL02960
322 CONTINUE	CAL02970
330 CONTINUE	CAL02980
C 340 WRITE(2,501)SYEAR,EYEAR	CAL02990
C 501 FORMAT(2X,'THE WORKING DAY TABLE STARTS FROM YEAR',16,' TO',16)	CAL03000
509 FORMAT(10X,315)	CAL03010
340 DO 600 I=1,31	CAL03020
DATE(I)=1	CAL03030
600 CONTINUE	CAL03040
DO 660 I=1,12	CAL03050
MON(I)=1	CAL03060
660 CONTINUE	CAL03070
YEARN=SYEAR	CAL03080
DO 610 K=1,10	CAL03090
WRITE(2,503)((HOLIDA(I,J,K),J=1,31,1),I=1,12,1)	CAL03100
C WRITE(6,503)((HOLIDA(I,J,K),J=1,31,1),I=1,12,1)	CAL03110
C WRITE(2,502)YEARN,DATE	CAL03120
YEARN=YEARN+1	CAL03130
610 CONTINUE	CAL03140
C 502 FORMAT(//,1X,'YEAR=',14,3113,12(1X,'MONTH=',12,1X,3113))	CAL03150
503 FORMAT(3112)	CAL03160
C 505 FORMAT(/,1X,'MONTH=',12,1X,3113)	CAL03170
C 502 FORMAT(//,1X,'YEAR=',14,3113,/,12(1X,'MONTH=',15,3113,/))	CAL03180
DO 50 I=1,100	CAL03190
IF(HOLID2(1,1).EQ.0)GOTO 60	CAL03200
IF(HOLID2(1,3).GT.0)GOTO 21	CAL03210
WRITE(4,10)(HOLID2(1,J),J=1,3)	CAL03220
C 10 FORMAT(/,2X,'YOU HAVE INPUT THE FIXED HOLIDAY FOR EACH YEAR ON',/,	CAL03230
C +10X,'MONTH=',15,5X,'DAY=',17)	CAL03240
10 FORMAT(/,3X,'YOU HAVE INPUT THE HOLIDAY ON',/,10X,'MONTH=',15,	CAL03250
+5X,'DAY=',15,5X,'YEAR=EVERY YEAR')	CAL03260
GOTO 50	CAL03270
21 WRITE(4,11)(HOLID2(1,J),J=1,3)	CAL03280
11 FORMAT(/,3X,'YOU HAVE INPUT THE HOLIDAY ON',/,10X,'MONTH=',15,	CAL03290
+5X,'DAY=',15,5X,'YEAR=',17)	CAL03300

50 CONTINUE	CAL0331C
60 WRITE(6,70)	CAL0332C
70 FORMAT(//,2X,'IF YOU WANT CHECK THE HOLIDAY YOU HAVE INPUT PLEASE	CAL0333C
+LOOK FILE "HOLIDAY DATE".',//)	CAL0334C
DO 80 I=1,31	CAL0335C
IMONTH(I)=1	CAL0336C
60 CONTINUE	CAL0337C
WRITE(6,499)	CAL0338C
499 FORMAT(//,2X,'IF YOU WANT THE WORKING DAY TABLE, YOU CAN PRINT FILE	CAL0339C
+ "WORKING DAY".',//)	CAL0340C
IYEAR=SYEAR	CAL0341C
DO 85 I=1,10	CAL0342C
WRITE(6,90)IYEAR	CAL0343C
90 FORMAT(//,10X,'YEAR',I10)	CAL0344C
WRITE(6,92)(IMONTH(IX),IX=1,31)	CAL0345C
92 FORMAT(/,15X,31I3)	CAL0346C
DO 88 J=1,12	CAL0347C
WRITE(6,91)J,(HOLIDA(J,KX,I),KX=1,31)	CAL0348C
91 FORMAT(/,2X,'MONTH=',I3,4X,31I3)	CAL0349C
68 CONTINUE	CAL0350C
IYEAR=IYEAR+1	CAL0351C
WRITE(6,93)	CAL0352C
93 FORMAT(/,'=====	CAL0353C
+=====')	CAL0354C
85 CONTINUE	CAL0355C
WRITE(1,36)SYEAR	CAL0356C
38 FORMAT(10X,15)	CAL0357C
STOP	CAL0358C
END	CAL0359C

```

C *****
C
C           MILES      FORTRAN
C           -----
C   MILES READ MILESTONE NODES INTERACTIVELY, AND ADD MILESTONE
C   ACTIVITIES AND DESIRED DATES AT THE END OF THE FILE BB DATA.
C   USER SHOULD GIVE NUMBER OF TOTAL DATA IN ORIGINAL BB DATA
C   AND MILESTONE NODES.
C   * WHEN THERE ARE MORE THAN 100 MILESTONES, CHANGE DIM. OF MILE
C
C *****
C   DIMENSION MILE(100,2),II(20000),MCATE(3),IDUT(4)
C   INTEGER HOLDY(120,31),YEAR,SMON,SDAY,SYR
C   COMMON/CONT/ YEAR,HOLDY,SMON,SDAY,SYR
C
C   DO 1 I=1,20000
C     READ(1,105,END=99) II(I)
C   1 CONTINUE
C
C   WRITE(6,301) I
301  FORMAT(/,' **  NUMBER OF BB DATA RECORDS IS MORE THAN ',15,
+/,,' ** CHECK PROGRAM')
99  NNUF=I-1
REWIND 1
C
C   WRITE(6,102)
102  FORMAT(/,' ENTER NUMBER OF MILESTONE NODES TO BE ADDED.')
C     READ(5,*) MNODE
C     IF(MNODE.LE.0) STOP
C     DO 10 I=1,MNODE
C       WRITE(6,103) I
103  FORMAT(' ENTER NODE OF MILESTONE ',13,' AND',/
+      ' , ' DESIRED MILESTONE DATE. ( MMDDYY OR 0 IF NONE, '
+      ' , ' E.G. 303,122585 OR 303,0')
C     READ(5,*) (MILE(I,J),J=1,2)
10  CONTINUE
C
C   WRITE(6,104)
104  FORMAT(/,' THANK YOU. IN PROCESSING ...')
C
C   TRANSLATE CALENDAR DATES INTO WORKING DATES
C
C     READ(3,100)YEAR
100  FORMAT(10X,15)
C     DO 195 I=1,120
C       READ(4,110)(HOLDY(I,J),J=1,31)
195  CONTINUE
110  FORMAT(31I2)
C     READ(7,*)SMON,SDAY,SYR
C
C     DO 15 I=1,MNODE
C       IF(MILE(I,2).EQ.0) GO TO 15
C       IYEF=MILE(I,2)/100
C       IMON=MILE(I,2)/10000

```

MILC001
MILC002
MILC003
MILC004
MILC005
MILC006
MILC007
MILC008
MILC009
MILC010
MILC011
MILC012
MILC013
MILC014
MILC015
MILC016
MILC017
MILC018
MILC019
MILC020
MILC021
MILC022
MILC023
MILC024
MILC025
MILC026
MILC027
MILC028
MILC029
MILC030
MILC031
MILC032
MILC033
MILC034
MILC035
MILC036
MILC037
MILC038
MILC039
MILC040
MILC041
MILC042
MILC043
MILC044
MILC045
MILC046
MILC047
MILC048
MILC049
MILC050
MILC051
MILC052
MILC053
MILC054
MILC055

```

MDATE(3)=MILE(1,2)-IYER*100+1900
MDATE(1)=IMON
MDATE(2)=(MILE(1,2)-IMON*10000-MDATE(3)+1900)/100
PRINT *,(MDATE(J),J=1,3)
CALL CALA(MDATE,MILE(1,2))
15  CONTINUE
C
READ(1,105,END=999) (II(I),I=1,NNUM)
105  FORMAT(A1)
C
END NOOE OF MILESTONE WILL BE C
ITC=0
DO 20 I=1,MNODE
    IOUT(4)=MOD(MILE(I,1),10)
    IOUT(3)=(MOD(MILE(I,1),100)-I1)/10
    IOUT(2)=(MOD(MILE(I,1),1000)-I2)/100
    IOUT(1)=(MOD(MILE(I,1),10000)-I3)/1000
20  WRITE(1,200) (IOUT(J),J=1,4),ITC,MILE(I,2)
200  FORMAT(15X,4I1,1X,15,15X,I3)
    WRITE(2,202) (MILE(I,1),I=1,MNODE)
202  FORMAT(20I4)
    STOP
C
999  WRITE(6,300) NNUM
300  FORMAT(/,' *** NUMBER OF BB DATA RECORDS IS LESS THAN ',15,
+/,', *** CHECK ')
    STOP
END
C
C
SUBROUTINE CALA(REPORT,REP)
C
C
INTEGER HCLDY(120,31),REP,YEAR,SMON,SCAY,SYR
INTEGER REPCRT(3),WD,MON,DA,YF
COMMON/CENT/ YEAR,HCLDY,SMON,SCAY,SYR
C
REPORT(2)=REPORT(2)-1
C
ISYF=(SYF-YEAR)*12+SMON
WD=C
IYF=(REPCRT(3)-YEAR)*12+REPCRT(1)
IF(IYF.LT.ISYF)GOTO 650
IF(IYF.EQ.ISYF)GOTO 530
DO 400 MI=ISYF,IYF
    IF(MI.EQ.ISYR)GOTO 410
    IF(MI.EQ.IYF)GOTO 420
    DO 440 MK=1,31
        WD=WD+HCLDY(MI,MK)
440  CONTINUE
    GOTO 400
420  LM=REPCRT(2)
    DO 430 ML=1,LM
        WD=WD+HCLDY(MI,ML)

```

MIL00560
MIL00570
MIL00580
MIL00590
MIL00600
MIL00610
MIL00620
MIL00630
MIL00640
MIL00650
MIL00660
MIL00670
MIL00680
MIL00690
MIL00700
MIL00710
MIL00720
MIL00730
MIL00740
MIL00750
MIL00760
MIL00770
MIL00780
MIL00790
MIL00800
MIL00810
MIL00820
MIL00830
MIL00840
MIL00850
MIL00860
MIL00870
MIL00880
MIL00890
MIL00900
MIL00910
MIL00920
MIL00930
MIL00940
MIL00950
MIL00960
MIL00970
MIL00980
MIL00990
MIL01000
MIL01010
MIL01020
MIL01030
MIL01040
MIL01050
MIL01060
MIL01070
MIL01080
MIL01090
MIL01100

FILE: MILES FORTRAN A VM/SF CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

430	CONTINUE	MIL0111
	GETC 400	MIL0112
410	DO 450 MJ=SDAY,31	MIL0113
	WD=WD+HOLDY(MI,MJ)	MIL0114
450	CONTINUE	MIL0115
400	CONTINUE	MIL0116
	GETC 470	MIL0117
580	LK=REPORT(2)	MIL0118
	IF(LK.LT.SDAY)GETC 650	MIL0119
	DO 460 LI=SDAY,LK	MIL0120
	WD=WD+HOLDY(IYR,LI)	MIL0121
460	CONTINUE	MIL0122
470	REP=WD	MIL0123
	GETC 500	MIL0124
C		MIL0125
650	PRINT *, ' ILLEGAL CALENDAR DATES ** PLEASE CHECK '	MIL0126
	REP = 0	MIL0127
500	CONTINUE	MIL0128
	PRINT *, ' CORRESPONDING WORKING DATE = ',REP	MIL0129
	RETURN	MIL0130
	END	MIL0131

FILE: RESLST FORTRAN A VM/SP CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

```
C ***** RES0001
C RES0002
C          RES0003
C          RES0004
C          RES0005
C          RES0006
C          RES0007
C          RES0008
C ***** RES0009
C          RES0100
C          RES0110
C          RES0120
C          RES0130
C          RES0140
C          RES0150
C          RES0160
C          RES0170
C          RES0180
C          RES0190
C          RES0200
C          RES0210
C          RES0220
C          RES0230
C          RES0240
C          RES0250
C          RES0260
C          RES0270
C          RES0280

          RESLST   FORTRAN
          -----
C    RESLST READS FILE BPS(SORTED BY SHOP+WORKCENTER) AND MAKES
C    A FILE BRES(SEQ. NUM, SHOP+WORKCENTER)
C    SWC          = SHOP+WORKCENTER INPUT
C    WC(500,2)    = SEQ., SHOP+WORKCENTER OUTPUT
C *****

      INTEGER WC(500,2),SWC,SW1
      I=0
      SW1=0
1    READ(1,100,END=10) SWC
      IF (SWC.EQ.SW1) GOTO 1
      I=I+1
      WC(I,1)=I
      WC(I,2)=SWC
      SW1=SWC
      GOTO 1
10   N=I
      DO 3 I=1,N
3     WRITE(2,101)(WC(I,J),J=1,2)
100  FORMAT(20X,15)
101  FORMAT(13,2X,15)
      STOP
      END
```

FILE: SHOPS FORTRAN A VM/SF CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

```
C *****
C
C          SHOPS      FORTRAN
C          -----
C      SHOPS READS FILE 9BRES, AND MAP EACH WORKCENTER TO THE
C      CORRESPONDING SEQUENTIAL SHOP NUMBER IN OUTPUT FILE BESHOPS,
C      AND ALSO CREATES OUTPUT FILE SSHPLST TO BE USED LATER FOR SSP.
C *****
C
C      DIMENSION INRES(250),MAP(250)
C
C      DO 10 I=1,250
C          READ(1,100,END=19) INRES(I)
100      FORMAT(5X,I5)
C      10 CONTINUE
C          PRINT *, 'DIMENSION IS NOT ENOUGH.. PLEASE CHECK.'
C          STOP
C      19 NUM=I-1
C          ISAVE=INRES(I)/100
C          NSAVE=1
C          MAP(I)=1
C
C      DO 20 I=2,NUM
C          IF ((INRES(I)/100).EQ.ISAVE) THEN
C              MAP(I)=NSAVE
C          ELSE
C              ISAVE=INRES(I)/100
C              NSAVE=NSAVE+1
C              MAP(I)=NSAVE
C          ENDIF
C      20 CONTINUE
C
C      WRITE(2,200) (MAP(I),I=1,NUM)
200      FORMAT(20I4)
C      WRITE(4,201) NUM
201      FORMAT('NUMBER OF WORKCENTER',I5)
C      WRITE(4,202) NSAVE
202      FORMAT('NUMBER OF SHOPS      ',I5)
C
C      ISAVE=INRES(1)/100
C      MAP(1)=ISAVE
C      NC=1
C      DO 30 I=2,NUM
C          IF (ISAVE.NE.(INRES(I)/100)) THEN
C              NC=NC+1
C              ISAVE=INRES(I)/100
C              MAP(NC)=ISAVE
C          ENDIF
C      30 CONTINUE
C
C      WRITE(3,202) (MAP(I),I=1,NC)
202      FORMAT(20I4)
C
C      STOP
```

FILE: SHOPS FORTRAN A VM/SF CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

END

SFC00560

```

C *****
C
C           INFILE1  FORTRAN
C           -----
C   INFILE1 READS FILES BBRES, BBSORT(SORTED BY I,J NCDE,SHOP+
C   WCRRCENT.), AND MAKES OUTPUT FILES BBALIST,EBWCLST,BBINTEN,
C   SEANUM. IT SETS THE TYPE OF ACTIVITIES, CALCULATES INTENSITIES
C   AND NUMBER OF SHOPS REQUIRED FOR EACH ACTIVITY.
C   INFILES AGGREGATES INFORMATIONS ON SHOP LEVEL.
C *****
C
C   INTEGER FROM,TO,SHOP,CENTER,DURAT,ACTLST(2500,5),ANUM(2500)
C   1 ,AN,WCACTV(2500,70),RESLST(200),NRES,FR,TT,RESNUM,INC,MHR
C   REAL INTLEV(2500,70)
C   COMMON/AAA/RESLST,NRES
C
C   DO 11 I=1,200
C   11  REAC(10,111,END=12)  RESLST(I)
C   12  NRES=I-1
C       IACTIV=C
C       KDUF=-10
C       FR=C
C       TT=C
C   1  READ(1,100,END=3) FROM ,TC,SHOP,CENTER,MHR,DURAT,ITYPE
C     LINE=LINE+1
C
C   100  FORMAT(15X,14,1X,15,1X,13,12,2X,15,2X,13,1X,11)
C   111  FORMAT(5),15)
C       IF(FROM.EQ.FR.AND.TO.EQ.TT) GOTO 2
C
C   A NEW ACTIVITY ; STORE ACTIVITY'S DATA
C   -----
C   KDUF=DURAT
C   FR=FROM
C   TT=TO
C   IACTIV=IACTIV+1
C   ANUM(IACTIV)=0
C   ACTLST(IACTIV,1)=IACTIV
C   ACTLST(IACTIV,2)=FROM
C   ACTLST(IACTIV,3)=TO
C   ACTLST(IACTIV,4)=DUFAT
C   IF(ITYPE.NE.1) ACTLST(IACTIV,5)=0
C   IF(ITYPE.EQ.1) ACTLST(IACTIV,5)=1
C
C   IF(DURAT.GT.0.AND.TO.NE.0) GOTO 35
C   ACTLST(IACTIV,5)=9
C   IF(10.EQ.0) ACTLST(IACTIV,5)=7
C   GOTO 1
C   33  INCCLD=9999
C       MHR1=C
C
C   SAME ACTIVITY ;
C   -----

```

INF00010
 INF00020
 INF00030
 INF00040
 INF00050
 INF00060
 INF00070
 INF00080
 INF00090
 INF00100
 INF00110
 INF00120
 INF00130
 INF00140
 INF00150
 INF00160
 INF00170
 INF00180
 INF00190
 INF00200
 INF00210
 INF00220
 INF00230
 INF00240
 INF00250
 INF00260
 INF00270
 INF00280
 INF00290
 INF00300
 INF00310
 INF00320
 INF00330
 INF00340
 INF00350
 INF00360
 INF00370
 INF00380
 INF00390
 INF00400
 INF00410
 INF00420
 INF00430
 INF00440
 INF00450
 INF00460
 INF00470
 INF00480
 INF00490
 INF00500
 INF00510
 INF00520
 INF00530
 INF00540
 INF00550

2	IWC=SHOP	INFC0560
	IF(IWC.NE.IWCOLD) GOTO 77	INFC0570
	MHR=MHR1+MHR	INFC0580
	MHR1=MHR	INFC0590
	GOTO 79	INFC0600
77	ANUM(IActiv)=ANUM(IActiv)+1	INFC0610
	AN=ANUM(IActiv)	INFC0620
	IWCOLD=IWC	INFC0630
	MHR1=MHR	INFC0640
C		INFC0650
C	REPLACE RESOURCE *CODE* BY ITS INDEX	INFC0660
	CALL SEARCH(IWC*100+CENTER,RESNUM)	INFC0670
	IF(FESNUM.NE.9999) GOTO 7	INFC0680
	PRINT *, 'ACTIVITY NUM=', IActiv, ' RESOURCE NOT IN LIST', IWC, CENTER	INFC0690
	PRINT *, '* LOGICAL ERROR... CHECK YOUR SOURCE	INFC0700
7	WCACTV(IActiv,AN)=RESNUM	INFC0710
78	IF(DURAT.NE.0) GOTO 8	INFC0720
	INTLEV(IActiv,AN)=0.	INFC0730
	GOTO 1	INFC0740
8	INTLEV(IActiv,AN)=FLOAT(MHR)/DURAT+.005	INFC0750
	GOTO 1	INFC0760
3	NACTIV=IActiv	INFC0770
C		INFC0780
	DO 4 I=1,NACTIV	INFC0790
	WRITE(2,200) (ACTLIST(I,J),J=1,5)	INFC0800
	IF(ANUM(I).EQ.0) GO TO 4	INFC0810
	WRITE(3,400) (WCACTV(I,J),J=1,ANUM(I))	INFC0820
	WRITE(4,300) (INTLEV(I,J),J=1,ANUM(I))	INFC0830
4	CONTINUE	INFC0840
	WRITE(7,400) ANUM	INFC0850
101	FORMAT(20I4)	INFC0860
200	FORMAT(I4,1X,I5,1X,I5,I7,2X,I1)	INFC0870
300	FORMAT(10F8.2)	INFC0880
400	FORMAT(20I4)	INFC0890
	STOP	INFC0900
	END	INFC0910
		INFC0920
		INFC0930
	SUBROUTINE SEARCH(CODE,LOCATN)	INFC0940
	INTEGER MAT(200),LENGTH,LOCATN,CODE	INFC0950
	COMMON/AAA/MAT,LENGTH	INFC0960
	L1=LENGTH	INFC0970
	LC=1	INFC0980
	IF(CODE.GT.MAT(L1)) GOTO 6	INFC0990
	IF(CODE.LT.MAT(1)) GOTO 6	INFC1000
	L=L1/2	INFC1010
10	IF(CODE.LT.MAT(L)) GOTO 1	INFC1020
	IF(CODE.GT.MAT(L)) GOTO 2	INFC1030
	LOCATN=L	INFC1040
	RETURN	INFC1050
1	IF(L.EQ.L1) GOTO 5	INFC1060
	L1=L	INFC1070
	L=(L1+L)/2	INFC1080
	IF(L.EQ.L1) L=L-1	INFC1090
	GOTO 10	INFC1100

FILE: INFILE1 FORTRAN A VM/SP CMS RELEASE 3.1.E 851112, CFC - L.C. BERKELEY

2 IF(L.EQ.L0) GOTO 5
L0=L
L=(L1+L)/2
IF(L.EQ.L0) L=L+1
GOTO 10

C CODE DOESN'T EXIT
6 LCCATN=9999
RETURN
END

.INFO1110
INFO1120
INFO1130
INFO1140
INFO1150
INFO1160
INFO1170
INFO1180
INFO1190
INFO1200

```

C *****
C
C          FOLLPREC  FORTRAN
C          -----
C  FOLLPREC PREPARES THE FOLLOWERS & PREDECESSORS TABLES
C  INPUT FILES  1 = 9BALIST (SORTED BY "FROM")
C               2 = 9BALTO  (SORTED BY "TO")
C  OUTPUT FILES 3 = 9BFOLL  (FOLLOWING ACTIVITIES)
C               4 = 9BPRED  (PREDECESSOR ACTIVITIES)
C               7 = 9BNET   (ACT.,NP,PPPOSITION,NF,FPPOSITION)
C *****
C
C  INTEGER FROM(3000,3),TO(3000,3),CCODE
C  INTEGER FOLL(10000),PRED(10000),NETINF(3000,4)
C  INTEGER TEMP(10000),NETEMP(3000,3),NPPOST,NPDSP,NPESF,NFROM,NTC
C
C  DO 8 I=1,10000
C    READ(1,100,END=9) (FROM(I,J),J=1,3)
100  FORMAT(14,1X,15,1X,15)
C    CONTINUE
C    9  LENGTH=I-1
C    READ(2,100) ((TO(K,J),J=1,3),K=1,LENGTH)
C
C    J1=1
C    NPPOST=0
C    IFLAG=0
C    NPDSP=0
C    NFROM=0
C    DO 10 I=2,LENGTH
C      IF (FROM(I,2).NE.FROM(I-1,2)) GOTO 2
C    GOTO 10
22  IFLAG=1
C
C  NEW SET OF ACTIVITIES, STARTING WITH A NEW "I"
C    2  CCODE=FROM(J1,2)
C    NFROM=NFROM+1
C    NETEMP(NFROM,1)=CCODE
C  FIND THE PREDECESSORS OF CODE (ACTIVITIES THAT END WITH "CODE")
C    CALL SEARCH(TO,CCODE,3,LOCATN,LENGTH)
C  LOCATN : FIRST OCCURANCE OF CODE. IF CODE DOESN'T EXIST LOCATN=9999
C    IF (LOCATN.NE.9999) GOTO 44
C    NPT=0
C    GOTO 55
C
C  STORE THE SET OF PREDECESSORS (TEMPORARY ARRAYS, SORTED BY "FROM")
C  A RECORD FOR EACH SET OF ACTIVITIES WHICH FOLLOWS THE SAME "I".
C
C  44  NT=1+NPPOST-LOCATN
C    DO 4 KK=LOCATN,LENGTH
C      IF (TO(KK,3).NE.CCODE) GOTO 5
C    4  TEMP(KK+NT)=TO(KK,1)
C    5  NPT=KK-LOCATN
C  55  NETEMP(NFROM,3)=NPPOST
C    NPPOST=NPPOST+NPT

```

NETEMP(NFROM,2) =NPT	FCLC056C
J1=1	FCLC057C
10 CONTINUE	FCLC058C
C WRITE THE LAST "SET" :	FCLC059C
IF(IFLAG.EQ.0) GOTO 22	FCLC060C
IFLAG=0	FCLC061C
PRINT *, 'FINISHED IDENTIFYING SET OF PREDECESSORS'	FCLC062C
C CONSTRUCT THE PREDECESSORS FILE:	FCLC063C
C -----	FCLC064C
NFROM=1	FCLC065C
DO 60 I=1,LENGTH	FCLC066C
IF(FROM(I,2).NE.NETEMP(NFROM,1))NFROM=NFROM+1	FCLC067C
NFT=NETEMP(NFROM,2)	FCLC068C
NPOST=NETEMP(NFROM,3)	FCLC069C
NETINF(I,4)=NPT	FCLC070C
NETINF(I,3)=NPOST	FCLC071C
DO 61 J=1,NPT	FCLC072C
61 PREC(NPCSP+J)=TEMP(NPOST+J)	FCLC073C
60 NPOST=NPOST+NPT	FCLC074C
PRINT *, 'FINISHED PREDECESSORS FILE'	FCLC075C
J1=1	FCLC076C
NPOST=0	FCLC077C
NPCSP=0	FCLC078C
NT0=0	FCLC079C
DO 110 I=2,LENGTH	FCLC080C
IF(TO(I,3).NE.TO(I-1,3)) GOTO 12	FCLC081C
GOTO 110	FCLC082C
122 IFLAG=1	FCLC083C
12 NT0=NT0+1	FCLC084C
CODE=TO(J1,3)	FCLC085C
NETEMP(NT0,1)=CODE	FCLC086C
C FIND THE FOLLOWERS OF ACTIVITIES WHICH END WITH "I".	FCLC087C
CALL SEARCH(FROM,CODE,2,LOCATN,LENGTH)	FCLC088C
IF(LOCATN.NE.9999) GOTO 144	FCLC089C
NFT=0	FCLC090C
GOTO 155	FCLC091C
C STORE SET OF FOLLOWERS WHICH END WITH "I"	FCLC092C
144 NT=1+NPOST-LOCATN	FCLC093C
DO 14 KK=LOCATN,LENGTH	FCLC094C
IF(FROM(KK,2).NE.CODE) GOTO 15	FCLC095C
14 TEMP(KK+NT)=FROM(KK,1)	FCLC096C
15 NFT=KK-LOCATN	FCLC097C
155 NETEMP(NT0,3)=NPOST	FCLC098C
NETEMP(NT0,2)=NFT	FCLC099C
NPOST=NPOST+NFT	FCLC100C
J1=1	FCLC101C
110 CONTINUE	FCLC102C
C WRITE THE "LAST" SET	FCLC103C
IF(IFLAG.EQ.0) GOTO 122	FCLC104C
LENT0=NT0	FCLC105C
PRINT *, 'FINISHED IDENTIFICATION OF SETS OF FOLLOWERS'	FCLC106C
C CONSTRUCT THE FOLLOWERS FILE	FCLC107C
	FCLC108C
	FCLC109C
	FCLC110C

```

C -----
  DC 160 I=1,LENGTH
  CALL SEARCH(NETEMP,FROM(1,3),1,NTC,LENTC)
  NFT=NETEMP(NTC,2)
  NPOST=NETEMP(NTC,3)
  NETINF(I,2)=NFT
  NETINF(I,1)=NPOST
  DC 161 J=1,NFT
161 FOLL(NPOST+J)=TEMP(NPOST+J)
160 NPOST=NPOST+NFT
  PRINT *, 'FINISHED PROCESSING. WRITING OUTPUT FILES'
  WRITE(3,200) FOLL
  WRITE(4,200) PRED
  WRITE(7,300) (I,(NETINF(I,J),J=1,4),I=1,LENGTH)
200 FORMAT(20I4)
300 FORMAT(15,4I5)
  STOP
  END
  SUBROUTINE SEARCH(MAT,CODE,COLUMN,LOCATN,LENGTH)
  -----
  C THIS SUB SEARCHES FOR "CODE" IN THE COLUMN "COLUMN" OF MATRIX
  C "MAT" WHERE THE MATRIX IS PRESORTED BY ASCENDING ORDER OF "COLUMN"
  C -----
  INTEGER MAT(3000,3),LENGTH,LOCATN,CODE,COLUMN

  L1=LENGTH
  LC=1
  IF(CODE.GT.MAT(L1,COLUMN)) GOTO 6
  IF(CODE.LT.MAT( 1,COLUMN)) GOTO 6
  L=LENGTH/2
10 IF(CODE.LT.MAT(L,COLUMN)) GOTO 1
  IF(CODE.GT.MAT(L,COLUMN)) GOTO 2
  LOCATN=L
  GOTO 3
  1 IF(L.EQ.L1) GOTO 5
  L1=L
  L=(L0+L1)/2
  IF(L.EQ.L1) L=L-1
  GOTO 10
  2 IF(L.EQ.L0) GOTO 6
  LC=L
  L=(L0+L1)/2
  IF(L.EQ.L0) L=L+1
  GOTO 10
  3 N=L-1
  DC 4 I=1,N
  IF(MAT(L-I,COLUMN).NE.CODE) GOTO 5
  4 CONTINUE
  5 LOCATN=L-I+1
C LOCATN IS THE FIRST OCCURANCE OF "CODE"
  RETURN
C "CODE" DOES NOT EXIST IN COLUMN
  6 LOCATN=9999
  RETURN

```

```

FOLL111C
FOLL112C
FOLL113C
FOLL114C
FOLL115C
FOLL116C
FOLL117C
FOLL118C
FOLL119C
FOLL120C
FOLL121C
FOLL122C
FOLL123C
FOLL124C
FOLL125C
FOLL126C
FOLL127C
FOLL128C
FOLL129C
FOLL130C
FOLL131C
FOLL132C
FOLL133C
FOLL134C
FOLL135C
FOLL136C
FOLL137C
FOLL138C
FOLL139C
FOLL140C
FOLL141C
FOLL142C
FOLL143C
FOLL144C
FOLL145C
FOLL146C
FOLL147C
FOLL148C
FOLL149C
FOLL150C
FOLL151C
FOLL152C
FOLL153C
FOLL154C
FOLL155C
FOLL156C
FOLL157C
FOLL158C
FOLL159C
FOLL160C
FOLL161C
FOLL162C
FOLL163C
FOLL164C
FOLL165C

```

FILE: FCLLPRED FORTRAN A VM/SF CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

END

FCL01660

```

C*****
C
C          RANKM      FORTRAN
C          -----
C      RANKM READS FILES BFOLL, EBPRD, EBNET(OUTPUTS FROM FOLLPRD)
C      AND GIVE SEQUENTIAL RANKS TO EACH ACTIVITIES BY RANKING FORWARD
C      THROUGH THE NETWORK. ALSO, CYCLES ARE IDENTIFIED BY COMPARING
C      FORWARD RANKS WITH BACKWARD RANKINGS.
C
C*****
C
C      INTEGER NCODE(2500,2)
C      INTEGER FOLL(10000),NF(2500),FPCS(2500),RANKF(2500)
C      INTEGER PRD(10000),NP(2500),PPCS(2500),RANKP(2500),FRANK(2500)
C      INTEGER NA,NCODE, NRANK, NFLAG, NPOSIT, NPRED, NPRD
C
C      READ (1,951,END=201) FOLL
201  READ (21,951,END=202) PRD
951  FCFMAT(2014)
202  DO 950 NA=1,10000
C      READ(9,120,END=1) (NCODE(NA,J),J=1,2)
950  READ(2,961,END=1) FPCS(NA), NF(NA), FPCS(NA), NP(NA)
961  FCFMAT(5X,415)
120  FCFMAT(4X,216)
1  NCODE = NA-1
C
C      ***RANKING FORWARD STARTS WITH UNPRECEDED ACTIVITIES.
C      NRANK=1
C      DO 2 NA=1,NCODE
C      IF(NP(NA).NE.0) GOTO 2
C      FRANK(NA)=NRANK
C      FRANK(NFRANK)=NA
C      NRANK=NRANK+1
2  CONTINUE
C
C      *** RANK FORWARD ***
C
10  NFLAG=0
C      DO 4 NA=1,NCODE
C      IF(FRANK(NA).GT.0) GOTO 4
C      NPRED=NF(NA)
C      NPOSIT=PPCS(NA)
C      DO 3 NPRED=1,NPRED
3  IF(FRANK(PRED(NPOSIT+NPRED)).EQ.0) GOTO 4
C      NFLAG=1
C      THERE ARE ADDITIONAL RANKINGS
C      FRANK(NA)=NRANK
C      FRANK(NFRANK)=NA
C      NRANK=NRANK+1
4  CONTINUE
C      IF(NFLAG.EQ.1) GOTO 10
C      NFLAG=1 IF THERE WERE ANY NEW RANKING DURING THE LAST LOOP.
C      WRITE(3,951) FRANK
C

```

RAN0001
RAN0002
RAN0003
RAN0004
RAN0005
RAN0006
RAN0007
RAN0008
RAN0009
RAN0010
RAN0011
RAN0012
RAN0013
RAN0014
RAN0015
RAN0016
RAN0017
RAN0018
RAN0019
RAN0020
RAN0021
RAN0022
RAN0023
RAN0024
RAN0025
RAN0026
RAN0027
RAN0028
RAN0029
RAN0030
RAN0031
RAN0032
RAN0033
RAN0034
RAN0035
RAN0036
RAN0037
RAN0038
RAN0039
RAN0040
RAN0041
RAN0042
RAN0043
RAN0044
RAN0045
RAN0046
RAN0047
RAN0048
RAN0049
RAN0050
RAN0051
RAN0052
RAN0053
RAN0054
RAN0055

C	START RANKING BACKWARD	RAN00560
C		RAN00570
	NRANK=-1	RAN00580
	DO 22 NA=1,NNODE	RAN00590
	IF (NF(NA).NE.0) GOTO 22	RAN00600
	RANKB(NA)=NRANK	RAN00610
	NRANK=NRANK-1	RAN00620
22	CONTINUE	RAN00630
110	NFLAG=0	RAN00640
	DO 44 NA=1,NNODE	RAN00650
	IF (RANKF(NA).LT.0) GOTO 44	RAN00660
	NPFED=NF(NA)	RAN00670
	NPCSI=FPCS(NA)	RAN00680
	DO 33 NFRD=1,NPRD	RAN00690
33	IF (RANKF(FCLL(NPCSI+NPRD)).EQ.0) GOTO 44	RAN00700
	NFLAG=1	RAN00710
	RANKB(NA)=NRANK	RAN00720
	NRANK=NRANK-1	RAN00730
44	CONTINUE	RAN00740
	IF (NFLAG.EQ.1) GOTO 110	RAN00750
	IRANK=NRANK+(-1)	RAN00760
	DO 11 NA=1,NNODE	RAN00770
	IF (RANKF(NA).EQ.0) GOTO 12	RAN00780
	RANKB(NA)=RANKF(NA)+IRANK	RAN00790
12	WRITE(7,101) NA, RANKF(NA), RANKB(NA), (NODES(NA,J),J=1,2)	RAN00800
	IF (RANKF(NA).EQ.0.AND.RANKB(NA).EQ.0) THEN	RAN00810
	WRITE(6,101) NA, RANKF(NA), RANKB(NA), (NODES(NA,J),J=1,2)	RAN00820
	WRITE(6,111)	RAN00830
	WRITE(7,111)	RAN00840
	ENDIF	RAN00850
11	CONTINUE	RAN00860
111	FORMAT(' ***** CYCLE OCCURRED *****')	RAN00870
101	FORMAT(3I10,5X,2I5)	RAN00880
	STOP	RAN00890
	END	RAN00900
		RAN00910

FILE: FTRAN FCFTRAN A VM/SP CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

```
C ***** FTR00010
C FTR00020
C FTRAN FORTRAN FTR00030
C ----- FTR00040
C FTRAN READS NODES AND COEFFICIENTS OF FLOW-TRANSFER AND CREATES FTR00050
C FILE B6TRC. B6TRC WILL BE INPUT TO CTRAN LATER. FTR00060
C FTR00070
C ***** FTR00080
C FTR00090
C READ INPUT FROM TERMINAL FTR00100
C FTR00110
C FTR00120
C FTR00130
C WRITE(6,*) ' ' FTR00140
1 WRITE(5,*) ' ENTER NODE AND COEFFICIENT OF FLOW-TRANSFER ' FTR00150
  WRITE(6,*) ' ( E.G. 303,0.5 FOR NODE 303 AND COEFFICIENT 0.5 ) ' FTR00160
  WRITE(6,*) ' END OF DATA; TYPE RETURN ONLY. ' FTR00170
  DO 10 I=1,100 FTR00180
    READ(6,*,END=15) NODE,COEF FTR00190
    IF(NODE.LT.0.OR.COEF.GT.1.OR.COEF.LT.0) THEN FTR00200
      WRITE(6,*) ' DATA ERROR.. PLEASE ENTER AGAIN. ' FTR00210
      GO TO 1 FTR00220
    ENDIF FTR00230
    WRITE(1,100) NODE,COEF FTR00240
10 CONTINUE FTR00250
15 WRITE(6,*) ' END OF DATA. THANK YOU...' FTR00260
100 FCFPAT(14,2X,F4.2) FTR00270
  STOP FTR00280
  END FTR00290
```

```

C *****
C
C          CTRAN      FORTRAN
C          -----
C      CTRAN READS FILES BRFOLL, BBPFED, BRNET, BBALIST, BBTKC,
C      AND SETS THE TRANSFER COEFFICIENTS ON THE FOLLOWERS AND
C      PREDECESSORS NODES(CTRANF,CTRANP) FOR EACH ACTIVITIES.
C      OUTFUT FILE IS EBTRAN.
C *****
C
      INTEGER FTEV(50),ST(10),CM(10),FOLL(10000),FRED(10000)
      INTEGER NF(2500),FPDS(2500),NF(2500),FPCS(2500)
      INTEGER START(2500),COMP(2500),N(10),ITYPE(2500)
      DIMENSION CTRANP(10000),CTRANF(10000),ALFA(50),AV(10)

C      READ THE NETWORK INFORMATION
      NAC=0
      READ(2,100)FOLL
      READ(12,100)PRED
100  FORMAT(20I4)
      DC 151 I=1,5000
      READ(3,200,END=152)FPCS(I),NF(I),FPCS(I),NP(I)
200  FORMAT(5X,4I5)
151  READ(4,300,END=152)START(I),COMP(I),ITYPE(I)
152  NNODES=I-1
      PRINT *, 'NUMBER OF NODES=',NNODES
300  FORMAT(5X,15,1X,15,9X,11)
      DC 161 I=1,1000
161  READ(1,400,END=162)FTEV(I),ALFA(I)
162  NC=I-1
400  FORMAT(14,2X,F4.2)
      PRINT *, 'NUM OF TRANSFER IS ',NC

C
      DC 81 I=1,10000
      CTRANP(I)=1.
81  CTRANF(I)=1.
      DC 42 K=1,NC
      DC 43 I=1,NNODES
      IF (COMP(I).NE.FTEV(K))GO TO 43
      IF (ITYPE(I).EQ.9) GO TO 43
      IF (ITYPE(I).EQ.2) GO TO 43
      IF (ITYPE(I).EQ.7) GO TO 43
      IS=FPCS(I)+1
      IE=IS+NF(I)-1
      DC 44 L=IS,IE
44  CTRANF(L)=ALFA(K)
43  CONTINUE
35  CONTINUE
      DC 45 I=1,NNODES
      IF (START(I).NE.FTEV(K))GO TO 45
      IF (ITYPE(I).EQ.9) GO TO 45
      IF (ITYPE(I).EQ.2) GO TO 45
      IF (ITYPE(I).EQ.7) GO TO 45

```

CTR0001
 CTR0002
 CTR0003
 CTR0004
 CTR0005
 CTR0006
 CTR0007
 CTR0008
 CTR0009
 CTR0010
 CTR0011
 CTR0012
 CTR0013
 CTR0014
 CTR0015
 CTR0016
 CTR0017
 CTR0018
 CTR0019
 CTR0020
 CTR0021
 CTR0022
 CTR0023
 CTR0024
 CTR0025
 CTR0026
 CTR0027
 CTR0028
 CTR0029
 CTR0030
 CTR0031
 CTR0032
 CTR0033
 CTR0034
 CTR0035
 CTR0036
 CTR0037
 CTR0038
 CTR0039
 CTR0040
 CTR0041
 CTR0042
 CTR0043
 CTR0044
 CTR0045
 CTR0046
 CTR0047
 CTR0048
 CTR0049
 CTR0050
 CTR0051
 CTR0052
 CTR0053
 CTR0054
 CTR0055

FILE: CTRAN FORTRAN A VM/SP CMS RELEASE 3.1.E 851112, CFC - L.C. BERKELEY

IS=FPS(I)+1	CTR00560
IE=IS+NP(I)-1	CTR00570
DC 46 L=IS,IE	CTR00580
46 CTRANP(L)=ALFA(K)	CTR00590
45 CONTINUE	CTR00600
42 CONTINUE	CTR00610
C	CTR00620
WRITE(7,211)CTRANF,CTRANP	CTR00630
211 FORMAT(2CF4.2)	CTR00640
C	CTR00650
STOP	CTR00660
END	CTR00670

C *****	CAPC001
C *	CAPC002
C * CAPDATE FORTRAN	CAPC003
C *	CAPC004
C * CAPDATE CALCULATES THE NUMBER OF WORKING DAYS	CAPC005
C * FROM THE PROJECT STARTING DATE TO CAPACITY CHANGE DATES	CAPC006
C *	CAPC007
C *****	CAPC008
INTEGER HOLDY(120,31), REP(100), YEAR, SMOH, S	CAPC009
INTEGER REPORT(100,3), MO, MON, DA, YF, SRF(100)	CAPC010
READ(1,100) YEAR	CAPC011
100 FORMAT(10X,15)	CAPC012
DC 195 I=1,120	CAPC013
READ(2,110)(HOLDY(I,J), J=1,31)	CAPC014
195 CONTINUE	CAPC015
110 FORMAT(3112)	CAPC016
C	CAPC017
READ(4,140) SMOH, SDAY, SYR	CAPC018
140 FORMAT(10X,315)	CAPC019
WRITE(6,130) SMOH, SDAY, SYR-1900	CAPC020
130 FORMAT(/, ' ENTER DATES WHEN SHOP CAPACITIES ARE EFFECTIVE ',	CAPC021
+ '(MM,DD,YY)',	CAPC022
+ /, ' FIRST EFFECTIVITY DATE IS THE STARTING DATE OF PROJECT; ',	CAPC023
+ 313, /, ' PLEASE ENTER ALL DATES AFTER THIS TIME WHEN SHOP ',	CAPC024
+ 'CAPACITIES CHANGE.', /,	CAPC025
+ ' IF END OF DATA, PLEASE PRESS RETURN ONLY.')	CAPC026
C	CAPC027
ISYF=(SYF-YEAR)*12+SMOH	CAPC028
REP(1)=0	CAPC029
K=1	CAPC030
DC 200 I=2,100	CAPC031
READ(5,*,END=300) MOH, DA, YR	CAPC032
DA=DA-1	CAPC033
YR=YR+1900	CAPC034
C	CAPC035
MO=0	CAPC036
IYR=(YR-YEAR)*12+MOH	CAPC037
IF(IYR.LT.ISYR) GOTO 650	CAPC038
IF(IYR.EQ.ISYR) GOTO 580	CAPC039
DC 400 MI=ISYR, IYR	CAPC040
IF(MI.EQ.ISYR) GOTO 410	CAPC041
IF(MI.EQ.IYF) GOTO 420	CAPC042
DC 440 MK=1,31	CAPC043
MO=MO+HOLDY(MI,MK)	CAPC044
440 CONTINUE	CAPC045
GOTO 400	CAPC046
420 LM=DA	CAPC047
DC 430 ML=1,LM	CAPC048
MO=MO+HOLDY(MI,ML)	CAPC049
430 CONTINUE	CAPC050
GOTO 400	CAPC051
410 DC 450 MU=SDAY,31	CAPC052
MO=MO+HOLDY(MI,MU)	CAPC053
450 CONTINUE	CAPC054
	CAPC055

FILE: CAPDATE FCFTRAN A VM/SF CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

400	CONTINUE	CAP00560
	GOTO 470	CAP00570
580	LK=CA	CAP00580
	IF(LK.LT.SDAY)GOTO 650	CAP00590
	DC 460 LI=SDAY,LK	CAP00600
	WD=WD+HCLDY(IYR,LI)	CAP00610
460	CONTINUE	CAP00620
470	K=K+1	CAP00630
	REP(K)=WD	CAP00640
	SRP(K,1)=MCM	CAP00650
	SRP(K,2)=DA+1	CAP00660
	SRP(K,3)=YR-1900	CAP00670
	GOTO 200	CAP00680
		CAP00690
650	WRITE(5,180)	CAP00700
180	FORMAT(' ** THE INPUT DATE IS BEFORE STARTING',	CAP00710
	+ ' DATE, PLS ENTER AGAIN.')	CAP00720
200	CONTINUE	CAP00730
C		CAP00740
300	WRITE(3,150)REP	CAP00750
150	FORMAT(20I4)	CAP00760
C		CAP00770
	DC 800 I=1,K	CAP00780
	IF(I.EQ.1) THEN	CAP00790
	ISYR=SYR-1900	CAP00800
	WRITE(9,801) SMCM,SDAY,ISYR	CAP00810
	GO TO 800	CAP00820
	ENDIF	CAP00830
	WRITE(9,801)(SRP(I,J),J=1,3)	CAP00840
801	FORMAT(3I2)	CAP00850
800	CONTINUE	CAP00860
	STOP	CAP00870
	END	CAP00880

```

C***** . SHC0001
C SHC0002
C          SHOFCAP  FORTRAN SHC0003
C          ----- SHC0004
C SHOFCAP READS SHOP NUMBER, NUMBER OF CAPACITY CHANGE AND CAPACITY SHC0005
C (MAN-HOUR) INTERACTIVELY, AND CREATES FILE BBECAP. SHC0006
C BBECAP WILL BE INPUT TO THE NEXT STEP PFECAP. SHC0007
C SHC0008
C***** SHC0009
C          READ INPUT FROM TERMINAL SHC0010
C SHC0011
C SHC0012
C SHC0013
C SHC0014
C SHC0015
C SHC0016
C SHC0017
C SHC0018
C SHC0019
C SHC0020
C SHC0021
C SHC0022
C SHC0023
C SHC0024
C SHC0025
C SHC0026
C SHC0027
C SHC0028
C SHC0029
C SHC0030
C
C          WRITE(6,*) ' '
C 1  WRITE(6,*) ' ENTER SHOP, NUMBER OF CAP. CHANGE AND CAPACITY.'
C    WRITE(6,*) ' ( E.G. 31,4,100 FOR SHOP 31, 4TH CHANGE, 100 MH )'
C    WRITE(6,*) ' ( IF NO CAP. BOUND FOR ALL SHOPS, ENTER 0,0,9999 )'
C    WRITE(6,*) ' ON END OF DATA, TYPE RETURN ONLY.'
C    DO 10 I=1,100
C      READ(6,*,END=15) ISHOP,NUM,CAP
C      IF(ISHOP.LT.0.OR.NUM.LT.0.OR.CAP.LT.0) THEN
C        WRITE(6,*) ' DATA ERROR.. PLEASE ENTER AGAIN.'
C        GO TO 1
C      ENDIF
C      WRITE(1,100) ISHOP,NUM,CAP
C      IF(ISHOP.EQ.0.AND.NUM.EQ.0.AND.CAP.EQ.9999) GO TO 15
C 10 CONTINUE
C 15 WRITE(6,*) ' END OF DATA. THANK YOU...'
C 100 FORMAT(13,2X,13,2X,F8.2)
C    STOP
C    END

```

```

C*****
C
C          PRECAP    FORTRAN
C          -----
C    PRECAP READ USER-DEFINED CAPACITY FILE CALLED BEBCAP , AND
C    MATCH THE SHOP ID. TO THE PROGRAM-DEFINED SEQUENCE CCDE, THEN
C    SETS THE CORRESPONDING CAPACITY IN THE FILE BEVCAF DATA.
C    NCHAN = NUMBER OF CAPACITY CHANGE
C    NSHCP = NUIMER OF SHOPS
C*****
C
C    DIMENSION CAP(30,250),IZ(10,20),NSEQ(250),KSHOP(250)
C    INTEGER ICHD(200)
C
C    READ(1,100) NFES,NSHCP
100  FORMAT(20X,15)
C
C    FIND OUT THE NUMBER OF CAPACITY CHANGES
C    II=0
C    DO 1 I=1,10
C    READ(2,103,END=2) (IZ(I,J),J=1,20)
103  FORMAT(20I4)
C    DO 7 J=1,20
C        II=II+1
C    7    ICHD(II)=IZ(I,J)
C    1    CONTINUE
C    2    NUM=I-1
C    DO 3 I=1,NUM
C    DO 3 J=1,10
C    IF(I.EQ.1.AND.J.EQ.1) GO TO 3
C    IF(IZ(I,J).EQ.0) GO TO 4
C    3    CONTINUE
C    4    NCHAN=(I-1)*20+J-1
C
C    READ(3,101) (NSEQ(I),I=1,NFES)
101  FORMAT(20I4)
105  FORMAT(16)
C    SAVE SHOP NUMBER
C    INIT=0
C    II=0
C    DO 5 I=1,NFES
C        READ(7,102) JSHP
C        IF(INIT.NE.NSEQ(I)) THEN
C            II=II+1
C            KSHOP(II)=JSHP
C            INIT=NSEQ(I)
C        ENDIF
C    5    CONTINUE
C    INITIALIZE CAP
C    DO 10 I=1,NCHAN
C    DO 10 J=1,NSHCP
10    CAP(I,J)=99999.
C
C    15  READ(4,*,END=99) ISHOP,ITER,CAFA

```

```

PRE0001
PRE0002
PRE0003
PRE0004
PRE0005
PRE0006
PRE0007
PRE0008
PRE0009
PRE0010
PRE0011
PRE0012
PRE0013
PRE0014
PRE0015
PRE0016
PRE0017
PRE0018
PRE0019
PRE0020
PRE0021
PRE0022
PRE0023
PRE0024
PRE0025
PRE0026
PRE0027
PRE0028
PRE0029
PRE0030
PRE0031
PRE0032
PRE0033
PRE0034
PRE0035
PRE0036
PRE0037
PRE0038
PRE0039
PRE0040
PRE0041
PRE0042
PRE0043
PRE0044
PRE0045
PRE0046
PRE0047
PRE0048
PRE0049
PRE0050
PRE0051
PRE0052
PRE0053
PRE0054
PRE0055

```

IF(ITER.LT.C.CR.ITER.GT.NCHAN) THEN	PREC056C
WRITE(6,*) ' * ILLEGAL NUMBER OF CHANGE IN BBFCAP. CHECK',ITER	PREC057C
STOP	PREC058C
ENDIF	PREC059C
IF(ITER.EQ.0) GO TO 99	PREC060C
C MAP THE SHOP CODE	PREC061C
REWIND 7	PREC062C
DC 20 I=1,250	PREC063C
C READ(7,102,END=29) JSHOP	PREC064C
102 FORMAT(6X,12)	PREC065C
IF(ISHOP.EQ.KSHOP(1)) THEN	PREC066C
C IPCS=NSEQ(1)	PREC067C
IPCS=1	PREC068C
GO TO 22	PREC069C
ENDIF	PREC070C
20 CONTINUE	PREC071C
C	PREC072C
29 WRITE(6,*) ' * ILLEGAL SHOP CODE. CHECK',ISHP	PREC073C
STOP	PREC074C
C	PREC075C
22 CAP(ITER,IPCS)=CAPA	PREC076C
GO TO 15	PREC077C
C	PREC078C
99 CONTINUE	PREC079C
DC 25 I=1,NCHAN	PREC080C
25 WRITE(9,200) (CAP(I,J),J=1,NSHCP)	PREC081C
200 FORMAT(10F8.2)	PREC082C
DC 30 I=1,NSHCP	PREC083C
DC 30 J=1,NCHAN	PREC084C
WRITE(10,204) KSHOP(1),ICHD(J),CAP(J,I)	PREC085C
30 CONTINUE	PREC086C
WRITE(9,202) NCHAN	PREC087C
204 FORMAT(14,1X,16,1X,F8.2)	PREC088C
202 FORMAT(' * CF CAPACITY CHANGE',15)	PREC089C
C	PREC090C
STOP	PREC091C
END	PREC092C

FILE: REPCATE FORTRAN A VM/SP CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

```
C *****
C *
C *          REPCATE  FORTRAN
C *          -----
C *          REPCATE  CALCULATE THE NUMBER OF WORKING DAYS
C *          FROM THE PROJECT STARTING DATE TO REPORTING DATES
C *
C *****
C
      INTEGER HOLDY(120,31),REP(100),YEAR,SMON,SDAY,SYR
      INTEGER REPCRT(100,3),WD,MON,DA,YR,SRF(100,3)
      READ(1,100)YEAR
100  FORCAT(1CX,15)
      DO 195 I=1,120
      READ(2,110)(HOLDY(I,J),J=1,31)
195  CONTINUE
110  FORMAT(31I2)
C
      READ(4,140)SMON,SDAY,SYR
140  FORMAT(1CX,3I5)
      WRITE(5,130)
130  FORCAT(/,1X,'ENTER THE REPORTING DATES ',
+ '( MM,DD,YY )',/,
+ ' IF END OF DATA, PLEASE PRESS RETURN TWICE.')
      DO 200 I=1,100
      READ(5,*,END=300)MON,DA,YR
      REPCRT(I,1)=MON
      REPCRT(I,2)=DA-1
      REPCRT(I,3)=YR+1900
      NI=I
200  CONTINUE
300  ISYF=(SYR-YEAR)*12+SMON
      DO 500 I=1,NI
      WD=0
      IYR=(REPCRT(I,3)-YEAR)*12+REPCRT(I,1)
      IF(IYR.LT.ISYR)GOTO 650
      IF(IYR.EQ.ISYR)GOTO 590
      DO 400 MI=ISYR,IYR
      IF(MI.EQ.ISYR)GOTO 410
      IF(MI.EQ.IYF)GOTO 420
      DO 440 MK=1,31
      WD=WD+HOLDY(MI,MK)
440  CONTINUE
      GOTO 400
420  LM=REPCRT(I,2)
      DO 430 ML=1,LM
      WD=WD+HOLDY(MI,ML)
430  CONTINUE
      GOTO 400
410  DO 450 MJ=SDAY,31
      WD=WD+HOLDY(MI,MJ)
450  CONTINUE
400  CONTINUE
      GOTO 470
580  LK=REPCRT(I,2)
      REPC0010
      REPC0020
      REPC0030
      REPC0040
      REPC0050
      REPC0060
      REPC0070
      REPC0080
      REPC0090
      REPC0100
      REPC0110
      REPC0120
      REPC0130
      REPC0140
      REPC0150
      REPC0160
      REPC0170
      REPC0180
      REPC0190
      REPC0200
      REPC0210
      REPC0220
      REPC0230
      REPC0240
      REPC0250
      REPC0260
      REPC0270
      REPC0280
      REPC0290
      REPC0300
      REPC0310
      REPC0320
      REPC0330
      REPC0340
      REPC0350
      REPC0360
      REPC0370
      REPC0380
      REPC0390
      REPC0400
      REPC0410
      REPC0420
      REPC0430
      REPC0440
      REPC0450
      REPC0460
      REPC0470
      REPC0480
      REPC0490
      REPC0500
      REPC0510
      REPC0520
      REPC0530
      REPC0540
      REPC0550
```

FILE: RECDATE FCFTRAN A VM/SF CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

```
      IF(LK.LT.SDAY)GOTO 650
      DC 460 LI=SDAY,LK
      WD=WD+MCLDY(1YR,LI)
460  CONTINUE
470  REP(I)=WD
      GOTO 500
650  WRITE(6,18C)1
180  FCRPAT(1X,'** THE INPUT DATE',13,2X,' IS BEFORE STARTING
      +DAY,PLS FEVISE IT',///)
      REP(I)=0
500  CONTINUE
C    WRITE(6,*)REP
      WRITE(3,15C)REP
150  FCRPAT(2014)
      DO 700 I=1,NI
          SRP(I,1)=REPORT(1,1)
          SRP(I,2)=REPORT(1,2)+1
          SRP(I,3)=MCD(REPORT(1,3),100)
700  CONTINUE
C
      STOP
      END
```

```
REPC0560
REPC0570
REPC0580
REPC0590
REPC0600
REPC0610
REPC0620
REPC0630
REPC0640
REPC0650
REPC0660
REPC0670
REPC0680
REPC0690
REPC0700
REPC0710
REPC0720
REPC0730
REPC0740
REPC0750
REPC0760
REPC0770
```

FILE: REPCON1 FORTRAN A VM/SP CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

```
C *****
C *
C *      REPCON1  FORTRAN
C *      -----
C *      REPCON1  READS BBTCAP AND BBTREP DATA AND MERGES TWO FILES
C *      TO SET REPORTING DATES INCLUDING CAPACITY CHANGE DATES.
C *
C *****
      INTEGER REP(100),CAP(100),SET(100)

C      READ CAPACITY CHANGE DATES AND REPORTING DATES
C
      READ(1,100) CAP
      READ(2,100) REP
100  FORMAT(20I4)
      DO 10 I=1,100
          IF(I.NE.1.AND.CAP(I).EQ.0) GO TO 11
          SET(I)=CAP(I)
10  CONTINUE
11  II=1
      DO 13 I=II,100
          J=I-II+1
          IF(J.NE.1.AND.REP(J).EQ.0) GO TO 15
          SET(I)=REP(J)
13  CONTINUE
15  III=I-1
      WRITE(3,200) (SET(I),I=2,III)
200  FORMAT(I4)
      STOP
      END
```

REPC001
REPC002
REPC003
REPC004
REPC005
REPC006
REPC007
REPC008
REPC009
REPC010
REPC011
REPC012
REPC013
REPC014
REPC015
REPC016
REPC017
REPC018
REPC019
REPC020
REPC021
REPC022
REPC023
REPC024
REPC025
REPC026
REPC027
REPC028
REPC029
REPC030
REPC031

```

C *****
C *
C *      REPCON2  FORTRAN
C *      -----
C *      REPCON2  READS BBBREP DATA AND CONVERTS IT TO CALENDAR
C *      TO SET REPORTING DATES INCLUDING CAPACITY CHANGE DATES.
C *
C *****
      INTEGER REF(100),SET(100)
      INTEGER WKD(100),YEAR,SMON,SDAY,SYR,WKDTB(120,31),CLND(100,3)

C      READ TEMPORARY REP. DATES IN BBBREP DATA
C
      DO 1 I=1,100
      READ(7,100,END=2) REP(I)
100  FORMAT(14)
      1  CONTINUE
      2  II=1-1
      ISAVE=REP(I)
      DO 10 I=2,II
      IF(ISAVE.NE.REP(I)) THEN
      ICNT=ICNT+1
      SET(ICNT)=ISAVE
      ISAVE=REP(I)
      ENDIF
      IF(I.EQ.II) THEN
      ICNT=ICNT+1
      SET(ICNT)=ISAVE
      ENDIF
10  CONTINUE
C
      WRITE(6,*) ' NUMBER OF REPORTING DATE IS',ICNT
      READ(11,*)
      WRITE(11,105) ICNT
105  FORMAT(' NUMBER OF REPORT DATES IS',15)
      WRITE(8,88) SET
      88  FORMAT(2014)
C
C      READ THE OUTPUT OF WORKING DATES
      REWIND 8
      READ(8,88,END=115)WKD
C      READ THE STARTING DATE OF THE PROJECT
115  READ(2,120)SMON,SDAY,SYR
120  FORMAT(10X,315)
C      READ THE YEAR OF BEGINNING IN THE WORKING TABLE
      READ(3,130)YEAR
130  FORMAT(10X,15)
C      READ THE WORKING DATES TABLE FROM FILE WORKING DATES
      DO 200 I=1,120
      READ(4,140)(WKDTB(I,J),J=1,31)
200  CONTINUE
140  FORMAT(3112)
      ISYF=(SYR-YEAR)*12+SMON
      DO 300 I=1,100

```

```

REPC001
REPC002
REPC003
REPC004
REPC005
REPC006
REPC007
REPC008
REPC009
REPC010
REPC011
REPC012
REPC013
REPC014
REPC015
REPC016
REPC017
REPC018
REPC019
REPC020
REPC021
REPC022
REPC023
REPC024
REPC025
REPC026
REPC027
REPC028
REPC029
REPC030
REPC031
REPC032
REPC033
REPC034
REPC035
REPC036
REPC037
REPC038
REPC039
REPC040
REPC041
REPC042
REPC043
REPC044
REPC045
REPC046
REPC047
REPC048
REPC049
REPC050
REPC051
REPC052
REPC053
REPC054
REPC055

```

```

M=WKD(I)
IF(P.EQ.C.AND.I.NE.1)GOTO 500
IF(P.EQ.0) THEN
  CLND(I,1)=SMON
  CLND(I,2)=SDAY
  CLND(I,3)=SYR-1900
  GC TO 300
ENDIF
DC 310 JI=SDAY,31
M=M-WKDTB(1SYR,J1)
JCUT=J1
IF(P.EQ.0)GOTO 380
310 CONTINUE
JSYR=1SYR+1
320 DC 330 JJ=1,31
M=M-WKDTB(JSYR,JJ)
JCUT=JJ
IF(P.EQ.0)GOTO 390
330 CONTINUE
JSYR=JSYR+1
GCTC 320
380 CLND(I,1)=SMON
CLND(I,2)=JCUT
CLND(I,3)=SYR-1900
GCTC 300
390 KMON=MOD(JSYR,12)
IF(KMON.GT.0)GOTO 395
KMON=12
395 KYR=(JSYR-KMON)/12+YEAR
CLND(I,1)=KMON
CLND(I,2)=JCUT
CLND(I,3)=KYR-1900
300 CONTINUE
C
500 DC 400 I=1,200
IF(CLND(I,1).EQ.0.AND.I.NE.1) GO TO 600
WRITE(9,15C) (CLND(I,J),J=1,3)
400 CONTINUE
150 FORPAT(12,12,12)
600 CONTINUE
STOP
END

```

```

REPC056
REPC057
REPC058
REPC059
REPC060
REPC061
REPC062
REPC063
REPC064
REPC065
REPC066
REPC067
REPC068
REPC069
REPC070
REPC071
REPC072
REPC073
REPC074
REPC075
REPC076
REPC077
REPC078
REPC079
REPC080
REPC081
REPC082
REPC083
REPC084
REPC085
REPC086
REPC087
REPC088
REPC089
REPC090
REPC091
REPC092
REPC093
REPC094
REPC095
REPC096
REPC097

```

```

C*****
C
C          COVERT    FORTRAN
C          -----
C    CCVERT READS FILES REWORK, BBALIST,
C    AND CREATES ACT. LIST INCLUDING REWORKS, SETS NEW RANKS OF
C    REWORKS, MAKES DURATIONS FILE AND PRECEDEILITIES FILE,
C    SETS FOLLOWERS, AND PREPARES ATTRIBUTES FILE FOR BRANCHING.
C*****
C
C    DIMENSION TATTR(250,5),ITRC(600)
C    DIMENSION PROCR(600,5),P1(600),P2(600),P3(600),P4(600),P5(600)
C    DIMENSION IST(600),ICP(600),PR(600),ISR(600),ICF(600)
C    INTEGER TFCLL(250,2),TYP(600),ITEST(100)
C    INTEGER DURR(600,5),D1(600),D2(600),D3(600),D4(600),D5(600)
C    INTEGER IS(2500),IC(2500),IDUR(2500),IT(2500),TRANK(2500)
C
C    READ REWORK DATA
C    DC 70 J=1,600
C    READ(2,80,END=71)IST(J),ICP(J),PR(J),ISR(J),ICR(J),TYP(J),
C    101(J),P1(J),D2(J),P2(J),D3(J),P3(J),D4(J),P4(J),D5(J),P5(J)
C    60 FCMPAT(A4,1X,A4,1X,F4.2,1X,A4,1X,A4,1X,I1,1X,5(1X,I3,1X,F4.2))
C    70 CONTINUE
C    PRINT *,* * NUMBER OF TEST ACTIVITIES IS GREATER THAN 600..*
C    STOP
C    71 NREF=J-1
C    SET TYPE OF TEST ACT., SAVE ACT. NUMBER AND CREATE TRANK
C    NTEST=0
C    K=0
C    DC 17 I=1,2500
C    READ(3,10,END=19)IS(I),IC(I),IDUR(I),IT(I)
C    DC 18 J=1,NREF
C    IF(IS(I).EQ.IST(J).AND.IC(I).EQ.ICP(J)) THEN
C    IT(I)=2
C    IF(ITEST(NTEST).NE.I) THEN
C    NTEST=NTEST+1
C    ITEST(NTEST)=I
C    ENDIF
C    ENDIF
C    18 CONTINUE
C    17 CONTINUE
C    10 FCMPAT(6X,A4,2X,A4,3X,I4,2X,I1)
C    PRINT *,* * NUMBER OF ACTIVITIES IS GREATER THAN 2500..CHECK*
C    19 NNODE=I-1
C    WRITE(5,*) * NUMBER OF TEST ACTIVITIES IS',NTEST
C    WRITE(4,100) ITEST
C    100 FCMPAT(10I5)
C    NACT=NNODE
C
C    DC 20 I=1,NNODE
C    IF(IT(I).NE.2) GO TO 20
C    K=K+1
C    TRANK(I)=K
C    20 CONTINUE
C    FINE TEST REPAIRS

```

CCVC001
 CCVC002
 CCVC003
 CCVC004
 CCVC005
 CCVC006
 CCVC007
 CCVC008
 CCVC009
 CCVC010
 CCVC011
 CCVC012
 CCVC013
 CCVC014
 CCVC015
 CCVC016
 CCVC017
 CCVC018
 CCVC019
 CCVC020
 CCVC021
 CCVC022
 CCVC023
 CCVC024
 CCVC025
 CCVC026
 CCVC027
 CCVC028
 CCVC029
 CCVC030
 CCVC031
 CCVC032
 CCVC033
 CCVC034
 CCVC035
 CCVC036
 CCVC037
 CCVC038
 CCVC039
 CCVC040
 CCVC041
 CCVC042
 CCVC043
 CCVC044
 CCVC045
 CCVC046
 CCVC047
 CCVC048
 CCVC049
 CCVC050
 CCVC051
 CCVC052
 CCVC053
 CCVC054
 CCVC055

KN=0	CEVC056C
DC 40 J=1,NREP	CEVC057C
IF(IS(I).EQ.IST(J).AND.IC(I).EQ.ICP(J)) GO TO 50	CEVC058C
40 CCNTINUE	CEVC059C
GC TO 20	CEVC060C
50 NACT=NACT+1	CEVC061C
IS(NACT)=ISR(J)	CEVC062C
IC(NACT)=ICR(J)	CEVC063C
IDUF(NACT)=0	CEVC064C
IF(TYP(J).NE.1) IT(NACT)=3	CEVC065C
IF(TYP(J).EQ.1) IT(NACT)=4	CEVC066C
M=NACT-NNOCE	CEVC067C
ITRC(M)=1	CEVC068C
DURF(M,1)=D1(J)	CEVC069C
DURF(M,2)=D2(J)	CEVC070C
DURF(M,3)=D3(J)	CEVC071C
DURF(M,4)=D4(J)	CEVC072C
DURF(M,5)=D5(J)	CEVC073C
PROER(M,1)=P1(J)	CEVC074C
PROER(M,2)=P2(J)	CEVC075C
PROER(M,3)=P3(J)	CEVC076C
PROER(M,4)=P4(J)	CEVC077C
PROER(M,5)=P5(J)	CEVC078C
IF(FR(J).LE.0.0)GO TO 60	CEVC079C
KN=KN+1	CEVC080C
TFOLL(K,KN)=NACT	CEVC081C
TATTR(K,KN+3)=PR(J)	CEVC082C
60 CCNTINUE	CEVC083C
J=J+1	CEVC084C
IF(IS(I).EQ.IST(J).AND.IC(I).EQ.ICF(J)) GO TO 50	CEVC085C
20 CCNTINUE	CEVC086C
PRINT #,'CALCULATING BRANCH PRBS'	CEVC087C
	CEVC088C
DC 90 J=1,K	CEVC089C
SUM=TATTR(J,4)+TATTR(J,5)	CEVC090C
TATTR(J,1)=1.-SUM	CEVC091C
TATTR(J,2)=9.99	CEVC092C
TATTR(J,3)=0.0	CEVC093C
IF(SUM.LE.0.) GO TO 90	CEVC094C
TATTR(J,4)=TATTR(J,4)/SUM	CEVC095C
TATTR(J,5)=TATTR(J,5)/SUM	CEVC096C
90 CCNTINUE	CEVC097C
C WRITE NEW LISTS	CEVC098C
DC 12 I=1,NACT	CEVC099C
12 WRITE(7,11)I,IS(I),IC(I),IDUR(I),IT(I)	CEVC100C
11 FCRPAT(14,2X,A4,2X,A4,3X,14,2X,11)	CEVC101C
WRITE(8,13)IRANK	CEVC102C
WRITE(9,13)DURR	CEVC103C
WRITE(10,14)PROBR	CEVC104C
WRITE(11,13)TFOLL	CEVC105C
WRITE(12,14)TATTR	CEVC106C
WRITE(13,13)ITRC	CEVC107C
13 FCRPAT(20I4)	CEVC108C
14 FCRPAT(20F4.2)	CEVC109C
C	CEVC110C

FILE: COVERT FORTRAN A VM/SF CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

STOP
END

CCV0111
CCV0112

C *****	RFFC0001
C	RFFC0002
C	RFFC0003
C	RFFC0004
C	RFFC0005
C	RFFC0006
C	RFFC0007
C	RFFC0008
C *****	RFFC0009
C	RFFC0010
C	RFFC0011
C	RFFC0012
C	RFFC0013
C	RFFC0014
C	RFFC0015
C	RFFC0016
C	RFFC0017
C	RFFC0018
C	RFFC0019
C	RFFC0020
C	RFFC0021
C	RFFC0022
C	RFFC0023
C	RFFC0024
C	RFFC0025
C	RFFC0026
C	RFFC0027
C	RFFC0028
C	RFFC0029
C	RFFC0030
C	RFFC0031
C	RFFC0032
C	RFFC0033
C	RFFC0034
C	RFFC0035
C	RFFC0036
C	RFFC0037
C	RFFC0038
C	RFFC0039
C	RFFC0040
C	RFFC0041
C	RFFC0042
C	RFFC0043
C	RFFC0044
C	RFFC0045
C	RFFC0046
C	RFFC0047
C	RFFC0048
C	RFFC0049
C	RFFC0050
C	RFFC0051
C	RFFC0052
C	RFFC0053
C	RFFC0054
C	RFFC0055

```

RPFCL      FORTRAN
-----
RPFCL READS FILES RPPAR, BALSTR, BBNET, BBFOLL, BBPFED, BTRCR
AND CREATES A NEW NETWORK INFO. FILE, FOLLOWERS, AND PREDECESSOR
FILE, INCLUDING REWORK ACTIVITIES.

*****

      INTEGER START(2500),COMP(2500)
      INTEGER DUR(2500),ITYPE(2500),ITRC(600)
      INTEGER FOLL(10000),NF(2500),FPOS(2500)
      INTEGER PRED(10000),NP(2500),PPOS(2500)

C
C      READ NETWORK FILE ( WITHOUT REWORKS )
      DC 172 I=1,2500
172 READ(3,131,END=173)L,FPOS(I),NF(I),FPOS(I),NF(I)
173 ITURN=I-1

C
      READ(4,141)FOLL
      READ(2,141)PRED
      READ(13,141)ITRC

C
C      READ ACTIVITY LIST ( INCLUDING REWORKS )
      DC 445 J=1,2500
445 READ(1,455,END=446)START(J),COMP(J),ITYPE(J)
455 FORMAT(6X,A4,2X,A4,9X,I1)
446 NNODE=J-1

C
C      CREATE FOLLOWER LIST
      IFF=FPOS(ITURN)+NF(ITURN)
      K=ITURN+1
      DC 51 I=K,NNODE
      N=0
      DC 52 J=1,NNODE
      IF (COMP(I).NE.START(J)) GO TO 52
      N=N+1
      FOLL(IFF+N)=J
52 CONTINUE
      NF(I)=N
      FPOS(I)=IFF
      IFF=IFF+NF(I)
51 CONTINUE

C
C      CREATE PREDECESSOR LIST
      IFF=PPOS(ITURN)+NP(ITURN)
      K=ITURN+1
      DC 61 I=K,NNODE
      N=0
      DC 62 J=K,NNODE
      IF (START(I).NE.COMP(J)) GO TO 62
      N=N+1
      PRED(IFF+N)=J
62 CONTINUE

```

FILE: RPFCL FORTRAN A VM/SP CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

```

NF(I)=N
PFOS(I)=IPP
IPP=IPP+NP(I)
61  CCNTINUE
   DC 161 I=1,NNODE
161  WRITE(7,131)I,PFOS(I),NF(I),PPCS(I),NF(I)
131  FORMAT(5I5)
   WRITE(8,141)FOLL
   WRITE(9,141)PRFD
141  FORMAT(2CI4)
   STOP
   END
```

```

RPF00560
RPF00570
RPF00580
RPF00590
RPF00600
RPF00610
RPF00620
RPF00630
RPF00640
RPF00650
RPF00660
RPF00670
```

```

C *****
C
C           RAB           FORTRAN
C           -----
C   RAB  READS FILES BBRES,BBFOLR,BALISTR,BEALIST,BNETR,BEWCLST,
C   BBINTEN AND BEANUM, AND CREATES NEW RESOURCE LIST, CALCULATES
C   INTENSITIES AND NUMBER OF SHOPS REQUIRED FOR EACH REWORKS.
C   STRCR AND KRRES ARE INPUT FILES, TOO.
C *****
C
C   INTEGER S(1000),C(1000),ST(1000),AI(1000),SHT(200)
C   INTEGER SANUM,SASHP(70),ITTAB(100)
C   INTEGER IS(2500),IC(2500),IST(2500),ICP(2500)
C   INTEGER FPCS(2500),ITYPE(2500),FOLL(10000),ITS(2500)
C   INTEGER ANUM(2500),ANUMM(2500),ASHOF(2500,70),ASHCPM(2500,70)
C   DIMENSION A(2500,70),AM(2500,70),SA(70)
C
C   FCONST=1.
C
C   READ IN THE ACTIVITY NUMBER OF TESTS FROM FILE BBTEST
C
C   READ(11,100) ITTAB
100  FORMAT(10I5)
C   READ THE DATA
C   PRINT *, 'READING FOLLOWERS'
C   READ(2,951) FOLL
951  FORMAT(20I4)
C   PRINT *, 'READING ACTIVITY-RESOURCE DATA'
C   READ(4,968,END=733) ANUMM
733  CONTINUE
C   DO 950 NA=1,2500
C   READ(7,952,END=953) IST(NA),ICP(NA)
952  FORMAT(5X,A5,1X,A5)
C   K=ANUMM(NA)
C   IF(K.EQ.0) GO TO 950
C   READ(9,962)(ASHCPM(NA,J),J=1,K)
962  FORMAT(20I4)
966  READ(10,963)(AM(NA,J),J=1,K)
963  FORMAT(10F8.2)
950  CONTINUE
C
953  N2=NA-1
C
C   DO 450 NA=1,2500
C   READ(17,452,END=453) IS(NA),IC(NA),ITYPE(NA),ITS(NA)
452  FORMAT(5X,A5,1X,A5,9X,11,3X,11)
454  READ(18,461) FPCS(NA)
461  FORMAT(5X,15)
450  CONTINUE
C
453  N3=NA-1
C
C   DO 230 J=1,1000

```

```

RAB0001C
RAE0002C
RAE0003C
RAE0004C
RAE0005C
RAE0006C
RAB0007C
RAB0008C
RAB0009C
RAB0010C
RAB0011C
RAB0012C
RAB0013C
RAB0014C
RAB0015C
RAE0016C
RAB0017C
RAB0018C
RAB0019C
RAB0020C
RAB0021C
RAB0022C
RAE0023C
RAE0024C
RAE0025C
RAB0026C
RAB0027C
RAB0028C
RAB0029C
RAB0030C
RAB0031C
RAE0032C
RAE0033C
RAB0034C
RAB0035C
RAB0036C
RAB0037C
RAB0038C
RAB0039C
RAB0040C
RAB0041C
RAB0042C
RAE0043C
RAE0044C
RAE0045C
RAE0046C
RAE0047C
RAB0048C
RAE0049C
RAE0050C
RAE0051C
RAE0052C
RAE0053C
RAE0054C
RAE0055C

```

230 READ(20,240,END=244)S(J),C(J),ST(J),A1(J)	RAB00560
240 FORMAT(14X,A5,A5,2X,14,1X,12)	RAB00570
244 NCL=J-1	RAB00580
DC 245 J=1,200	RAB00590
245 READ(21,241,END=246) SHT(J)	RAB00600
241 FORMAT(5X,15)	RAB00610
246 NRT=J-1	RAB00620
C	RAB00630
DC 20 I=1,N3	RAB00640
IF(IITYPE(I).EQ.3) GO TO 120	RAB00650
IF(IITYPE(I).EQ.4) GO TO 120	RAB00660
IF(IITYPE(I).EQ.9) GO TO 60	RAB00670
IF(IITYPE(I).NE.2) GO TO 80	RAB00680
IF(IC(I).LE.9999) GO TO 80	RAB00690
ISTART=IS(I)	RAB00700
K=FFGS(I)+1	RAB00710
J=FCLL(K)	RAB00720
ICOMP=IC(J)	RAB00730
GO TO 90	RAB00740
80 ICOMP=IC(I)	RAB00750
ISTART=IS(I)	RAB00760
90 CONTINUE	RAB00770
DC 30 K=1,N2	RAB00780
30 IF (ISTART.EQ.IST(K).AND.ICOMP.EQ.ICP(K))GO TO 40	RAB00790
C	RAB00800
PLT STANDARD INPUT MIX AND AMOUNTS TO THE TEST	RAB00810
CCC IF(ITS(I).EQ.0)ITS(I)=1	RAB00820
CCC K=ITAB(ITS(I))	RAB00830
K=I	RAB00840
C	RAB00850
40 ANUM(I)=ANLMM(K)	RAB00860
C	RAB00870
PRINT *,I=,I	RAB00880
NRS=ANUM(I)	RAB00890
DC 50 L=1,NRS	RAB00900
ASHCP(I,L)=ASHCPM(K,L)	RAB00910
50 A(I,L)=AM(K,L)	RAB00920
GO TO 20	RAB00930
60 ANUM(I)=0	RAB00940
GO TO 20	RAB00950
120 M=0	RAB00960
ANUM(I)=0	RAB00970
DC 140 J=1,NCL	RAB00980
IF (IS(I).NL.S(J).OR.IC(I).NE.C(J))GO TO 140	RAB00990
C	RAB01000
PRINT *,S(J),C(J),ST(J),A1(J)	RAB01010
DC 150 L=1,NRT	RAB01020
IF (ST(J).EQ.SHT(L)) GO TO 160	RAB01030
150 CONTINUE	RAB01040
IF(ST(J).NE.0) PRINT *, ' SHUP DOES NOT EXIST FOR ',I,ST(J)	RAB01050
GO TO 140	RAB01060
160 ANUM(I)=ANUM(I)+1	RAB01070
M=M+1	RAB01080
	RAB01090
	RAB01100

FILE: RAB FCFTRAN A VM/SF CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

ASHCP(I,P)=L	RAE0111
A(I,M)=A1(J)	RAE0112
140 CONTINUE	RAE0113
20 CONTINUE	RAE0114
C WRITE NEW INPUT LISTS	RAE0115
DC 4 I=1,N3	RAE0116
IF(ANLM(I).EQ.0) GO TO 4	RAE0117
WRITE(3,400)(ASHCP(I,J),J=1,ANUM(I))	RAE0118
WRITE(14,300)(A(I,J),J=1,ANUM(I))	RAE0119
4 CONTINUE	RAE0120
WRITE(1,400) ANUM	RAE0121
300 FCMPAT(1CF8.2)	RAE0122
400 FCMPAT(2014)	RAE0123
STOP	RAE0124
END	RAE0125

```

C ***** MIC00010
C MIC00020
C MICRG FORTRAN MIC00030
C ----- MIC00040
C MICRG DOES SCHEDULING ACCORDING TO 3 CATEGORIES ; MIC00050
C 1. DETERMINISTIC SCHEDULING MIC00060
C 2. STOCHASTIC SCHEDULING REGARDING TEST NETWORK LOOPS MIC00070
C 3. SIMULATION USING RANDOM RESOURCE REQUIREMENTS MIC00080
C ALL 3 CAT. CAN USE UPGRADING OR UP & DOWNGRADING ALGORITHM MIC00090
C * VERSION FOR INTERACTIVE RUN MIC00100
C ***** MIC00110
MIC00120
CHAFACTER#50 NAME MIC00130
INTEGER TOTMHR(50), TMILE(50), ACSHP(50), ACCWC(200), MIC00140
+ TOTMDY(300), IMILE(3), SMILE(50), MDATE(3) MIC00150
REAL RESREQ(50), ATOT(300) MIC00160
DIMENSION A(2500,70), ITYPE(2500) MIC00170
INTEGER ASHP(2500,70), ANUM(2500), MILE(2500), MILSTN(50) MIC00180
INTEGER FROM(2500), TU(2500) MIC00190
DIMENSION ZL(2500), ZU(2500), Z(2500) MIC00200
DIMENSION CAP(200), AR(200), LOST(15), DSEEDS(5), A1(2500,70) MIC00210
DIMENSION PCV(2500), TLU(2500) MIC00220
DIMENSION TSTART(2500), TFINIS(2500), ATFIN(2500) MIC00230
DIMENSION CIC(50,200) MIC00240
REAL LS(2500), DFL(30), ACUMRS(200), ACLMSP(50) MIC00250
INTEGER FOLL(10000), NF(2500), FPOS(2500), WCSHCP(200), NSHCP MIC00260
INTEGER PRED(10000), NP(2500), PPOS(2500), REPEAT(50), NREPT MIC00270
REAL CTRANF(10000), CTRANP(10000) MIC00280
INTEGER PRI(2500), DUR(2500), RANK(2500), TFCC(50) MIC00290
INTEGER STATUS(2500), IACTIV(1000), IREADY(1000) MIC00300
DIMENSION NPRI(1000), DRV(1000), IREV(1000), TEVENT(10000) MIC00310
INTEGER CUDE, TIMVEC(10000,2), ADTIME(1000,11) MIC00320
INTEGER TFCLL(250,2), TRANK(2500), DURR(600,5) MIC00330
DIMENSION TATTR(250,5), PRCB(10), RDUR(10), EDUR(2500), PROER(600,5) MIC00340
INTEGER YEAR, SMON, SDAY, SYR, WKDTB(120,31), CALND1(3), CALND2(3) MIC00350
MIC00360
COMMON/BB1/ IREADY, IACTIV, NREADY, NACTIV MIC00370
COMMON/BB3/ JNX, JF, JL, LWC MIC00380
COMMON/BB4/ CTRANF, CTRANP MIC00390
COMMON/BB5/ LS, NNODE, RANK, ITURN MIC00400
COMMON/BB6/ PRED, NP, PPOS MIC00410
COMMON/BB7/ FOLL, NF, FPOS MIC00420
COMMON/BB8/ PCV, TLU, TFINIS, Z, CLOCK, DUR, LENGTH MIC00430
COMMON/BB9/ ANUM, ASHP, A, AR MIC00440
COMMON/BB10/ IREV, NPRI, JFLAG MIC00450
COMMON/BB11/ ZL MIC00460
COMMON/BB12/ ITYPE, ZU, CAP, TFCC, CIC MIC00470
COMMON/BB13/ NRES, NCC MIC00480
COMMON/CC9/ WCSHCP, NSHCP MIC00490
COMMON/CC10/ ACUMRS, ACUMSP MIC00500
COMMON/RMVMN/ TEND MIC00510
COMMON/YYY/ LTIME MIC00520
COMMON/TTT/ TEVENT, TIMVEC, ADTIME, NXTVEC MIC00530
COMMON/DDON/ SMON, SDAY, SYR, YEAR, WKDTB MIC00540
CHAFACTER#1 NTES MIC00550

```

C	READ INPLT FILES	MIC0056C
C		MIC0057C
	READ(33,322) NRES,NSHOP	MIC0058C
	READ(34,322) NCC	MIC0059C
	READ(35,322) NREPD	MIC0060C
C		MIC0061C
	322 FORMAT(20X,15)	MIC0062C
C	READ SEED FOR RANDOM NUMBER	MIC0063C
	READ(17,1777,END=777) NRAND	MIC0064C
	GO TO 778	MIC0065C
	777 WRITE(6,*) ' PLEASE ENTER A NUMBER FOR RANDOM NUMBER GENERATION'	MIC0066C
	READ(5,*) NRAND	MIC0067C
	778 CONTINUE	MIC0068C
C		MIC0069C
	WRITE(6,*) ' *****'	MIC0070C
	WRITE(6,*) ' # M I C O R G #'	MIC0071C
	WRITE(6,*) ' *****'	MIC0072C
	WRITE(6,*) ' '	MIC0073C
		MIC0074C
		MIC0075C
	WRITE(6,*) ' ENTER OPTION 0 = READ PARAMETERS FROM INPUT FILE'	MIC0076C
	WRITE(6,*) ' 1 = INTERACTIVE INPUT'	MIC0077C
	READ(5,*) I1111	MIC0078C
	IF(I1111.EQ.0) THEN	MIC0079C
	READ(1,323) NAME,	MIC0080C
	+ IRAPOL,ILOW,IUP,NTEST,KTEND,NCYCLE,ITARG,ICALD	MIC0081C
	IF(NCYCLE.EQ.1) ISIM=1	MIC0082C
	IF(NCYCLE.NE.1) ISIM=2	MIC0083C
	GO TO 342	MIC0084C
	ENDIF	MIC0085C
	WRITE(6,*) ' ENTER PROJECT NAME (MAX. 50 CHARACTERS)'	MIC0086C
	WRITE(6,*) ' -----'	MIC0087C
	READ(5,333) NAME	MIC0088C
333	FORMAT(A50)	MIC0089C
	WRITE(6,*) ' ENTER ASSIGNMENT POLICY :'	MIC0090C
	1 WRITE(6,*) ' 1 = UPGRADING ONLY, 2 = UP & DOWNGRADING'	MIC0091C
	READ(5,*) IRAPOL	MIC0092C
	IF(IRAPOL.LT.1.OR.IRAPOL.GT.2) THEN	MIC0093C
	WRITE(6,*) ' * ILLEGAL POLICY. PLEASE ENTER AGAIN.'	MIC0094C
	GO TO 1	MIC0095C
	ENDIF	MIC0096C
		MIC0097C
	WRITE(6,*) ' '	MIC0098C
	WRITE(6,*) ' ENTER LOWER & UPPER BOUND OF INTENSITY :'	MIC0099C
2	WRITE(6,*) ' (USE PERCENTAGE VALUE E.G. 100 FOR 100 %)'	MIC0100C
	READ(5,*) ILOW,IUP	MIC0101C
	IF(ILOW.LE.0.OR.IUP.GT.999) THEN	MIC0102C
	WRITE(6,*) ' * ILLEGAL BOUND. PLEASE ENTER AGAIN.'	MIC0103C
	GO TO 2	MIC0104C
	ENDIF	MIC0105C
		MIC0106C
	WRITE(6,*) ' '	MIC0107C
	WRITE(6,*) ' ARE THERE ANY TEST ACTIVITIES ? (Y OR N)'	MIC0108C
	READ(5,334) NNTES	MIC0109C
		MIC0110C

334	FORMAT(A1)	MIC01110
	IF(NTES.EQ.'Y ') NTEST=1	MIC01120
	IF(NTES.EQ.'N ') NTEST=0	MIC01130
	WRITE(6,*) ' '	MIC01140
	WRITE(6,*) ' ENTER NUMBER OF RUN CYCLES : '	MIC01150
4	WRITE(6,*) ' 1 = SINGLE RUN WITH DETERMINISTIC RESOURCE USE '	MIC01160
	WRITE(6,*) ' N = N SIMULATIONS WITH RANDOM RESOURCE USE '	MIC01170
	READ(5,*) NCYCLE	MIC01180
	IF(NCYCLE.LT.1) THEN	MIC01190
	WRITE(6,*) ' * ILLEGAL NUMBER. PLEASE ENTER AGAIN. '	MIC01200
	GO TO 4	MIC01210
	ENDIF	MIC01220
	IF(NCYCLE.EQ.1) THEN	MIC01230
	ISIM=1	MIC01240
	KTEND=9999	MIC01250
	GO TO 6	MIC01260
	ENDIF	MIC01270
	ISIM=2	MIC01280
	WRITE(6,*) ' '	MIC01290
	WRITE(6,*) ' ENTER LIMIT OF PROJECT FINISH TIME FOR SIMULATIONS '	MIC01300
	READ(5,*) KTEND	MIC01310
		MIC01320
		MIC01330
6	WRITE(6,*) ' '	MIC01340
	WRITE(6,*) ' ENTER PROJECT TARGET FINISH TIME (0 IF NONE) '	MIC01350
	READ(5,*) ITARG	MIC01360
	WRITE(6,*) ' '	MIC01370
	WRITE(6,*) ' CHOOSE ONE OPTION FOR REPORT DATES : '	MIC01380
	WRITE(6,*) ' 0 = REPORT IN WORKING DATES '	MIC01390
	WRITE(6,*) ' 1 = REPORT IN CALENDAR DATES '	MIC01400
	READ(5,*) ICALD	MIC01410
342	CONTINUE	MIC01420
	WRITE(6,*) ' THANK YOU. IN PROCESSING..... '	MIC01430
C		MIC01440
	ZLOW=ILOW*.01	MIC01450
	ZUP=IUP*.01	MIC01460
	TIME=0.	MIC01470
	TEND=KTEND*.1.	MIC01480
C		MIC01490
	WRITE(6,*) ' READING FOLL & PRED '	MIC01500
	READ(2,951) FOLL	MIC01510
	READ(3,951) PRED	MIC01520
	WRITE(6,*) ' READING ACTIVITY-RESOURCE DATA '	MIC01530
	READ(4,951) ANUM	MIC01540
	WRITE(6,*) ' READING TRANSFER COEFFICIENTS '	MIC01550
C	INITIALIZE COEFFICIENTS TO 1.	MIC01560
	DO 344 I=1,10000	MIC01570
	CTRANF(I)=1.	MIC01580
	CTRANP(I)=1.	MIC01590
344	CONTINUE	MIC01600
	READ(12,953,END=7) CTRANF,CTRANP	MIC01610
7	WRITE(6,*) ' READING ACTIVITIES DATA '	MIC01620
	NMILE=0	MIC01630
	ITURN=0	MIC01640
	DO 950 NA=1,2500	MIC01650

READ(7,952,END=101) FROM(NA),TO(NA),DLR(NA),ITYPE(NA)	MIC0166C
IF(ITYPE(NA).NE.3.AND.ITYPE(NA).NE.4) ITURN=ITURN+1	MIC0167C
IF(ITYPE(NA).NE.7) GOTO 954	MIC0168C
NMILE=NMILE+1	MIC0169C
MILE(NA)=NMILE	MIC0170C
MILSTN(NMILE)=FROM(NA)	MIC0171C
954 READ(8,961) FPOS(NA), NF(NA), PPOS(NA), NP(NA)	MIC0172C
K=ANUM(NA)	MIC0173C
IF(K.EQ.0) GO TO 950	MIC0174C
READ(9,951) (ASHOP(NA,J),J=1,K)	MIC0175C
READ(10,963) (AI(NA,J),J=1,K)	MIC0176C
950 CONTINUE	MIC0177C
101 NNODE=NA-1	MIC0178C
C	MIC0179C
C READ MAPPING WC >> SHOP.	MIC0180C
C READ(13,951) (WCSHOP(I),I=1,NRES)	MIC0181C
C READ DATES OF REPORTING DATES	MIC0182C
C IF(NREPT.GT.0)	MIC0183C
C * READ(16,951) (REPDAT(I),I=1,NREPT)	MIC0184C
C	MIC0185C
C IF THERE IS NO TEST ACT. SKIP READING	MIC0186C
C IF(NTEST.EQ.0) GOTO 379	MIC0187C
C WRITE(6,*) ' READING REWCRC-ACTIVITIES DATA'	MIC0188C
C READ(18,951) TFOLL	MIC0189C
C READ(19,953) TATTR	MIC0190C
C READ(30,951) DURR	MIC0191C
C READ(31,953) PROBR	MIC0192C
C READ(20,951) TRANK	MIC0193C
379 CONTINUE	MIC0194C
C WRITE(6,*) ' READING CAPACITY(TIME,RESOURCE)'	MIC0195C
C DC 497 I=1,NCC	MIC0196C
C READ(14,3001,END =497) (CIC(I,J),J=1,ASHOP)	MIC0197C
497 CONTINUE	MIC0198C
C IF(NCC.LE.1) GOTO 387	MIC0199C
C READ TIME OF CAPACITY CHANGES.	MIC0200C
C READ (15,951,END=387) (TFCC(I),I=1,NCC)	MIC0201C
387 CONTINUE	MIC0202C
C WRITE(6,*) ' READING RANK OF ACTIVITIES'	MIC0203C
C READ(11,951) (RANK(I),I=1,NNODE)	MIC0204C
C	MIC0205C
C PREPARE CONVERTING WORKING DATES INTO CALENDAR DATES	MIC0206C
C AND VICE VERSA	MIC0207C
C READ THE STARTING DATE OF THE PROJECT	MIC0208C
C READ(52,520)SMON,SDAY,SYR	MIC0209C
520 FCRPAT(10X,315)	MIC0210C
C READ THE YEAR OF BEGINNING IN THE WORKING TABLE	MIC0211C
C READ(53,530)YEAR	MIC0212C
530 FCRPAT(10X,15)	MIC0213C
C READ THE WORKING DATES TABLE FROM FILE WORKING DATES	MIC0214C
C DC 533 I=1,120	MIC0215C
C READ(54,540)(WKDTB(I,J),J=1,31)	MIC0216C
533 CONTINUE	MIC0217C
540 FORMAT(31I2)	MIC0218C
C	MIC0219C
C TRANSLATE CALENDAR DATE OF TARGET FINISH INTO WORKING DAY	MIC0220C

C	IF(ITARG.EQ.0) GO TO 551	MIC02210
	IYER=ITARG/100	MIC02220
	IMON=ITARG/10000	MIC02230
	MDATE(3)=ITARG-IYER*100+1900	MIC02240
	MDATE(1)=IMON	MIC02250
	MDATE(2)=(ITARG-IMON*10000-MDATE(3)+1900)/100	MIC02260
	CALL CALA(MDATE,ISCFIN)	MIC02270
551	CONTINUE	MIC02280
C		MIC02290
C		MIC02300
C	END OF DATA	MIC02310
C	-----	MIC02320
C	WRITE(6,*) ' END OF DATA . INITIALIZE VALUES FOR SIMULATION '	MIC02330
C	-----	MIC02340
C		MIC02350
C	FIND EXPECTED DURATIONS FOR TEST ACTIVITIES	MIC02360
C		MIC02370
C	IF (ITURN .EQ. NNODE) GO TO 2999	MIC02380
C	REPAIR ACTIVITIES	MIC02390
	K=ITURN +1	MIC02400
	DO 84 I=K,NNODE	MIC02410
	M=I-ITURN	MIC02420
	DO 83 N=1,5	MIC02430
	PROB(N)=PROBR(M,N)	MIC02440
83	RDUR(N)=CURR(M,N)	MIC02450
84	DUR(I)=XPEAN(PROB,RDUR,5)	MIC02460
2999	NFLAG=0	MIC02470
	DO 383 I=1,NNODE	MIC02480
	IF(ITYPE(I).NE.2) GOTO 381	MIC02490
C	FIND EXPECTED DURATIONS OF REPAIR AFTER TEST	MIC02500
	J=TRANK(I)	MIC02510
	IF(TFOLL(J,1).EQ.0.AND.TFOLL(J,2).EQ.C) ITYPE(I)=C	MIC02520
	IF(ITYPE(I).NE.2) GOTO 381	MIC02530
	DO 384 L=1,2	MIC02540
	NC=0	MIC02550
	PROB(L)=TATTR(J,3+L)	MIC02560
	K=TFOLL(J,L)	MIC02570
	RDUR(L)=0.	MIC02580
	IF(K.EQ.C) GOTO 384	MIC02590
	RDUR(L)=DUR(K)	MIC02600
385	IS=FPOS(K)+1	MIC02610
	M=FOLL(IS)	MIC02620
	NC=NC+1	MIC02630
	IF(TO(I).EQ.TO(K)) GOTO 384	MIC02640
	K=FOLL(IS)	MIC02650
	RDUR(L)=RDUR(L)+DUR(K)	MIC02660
	IF(NC.LE.4) GOTO 385	MIC02670
C		MIC02680
	WRITE (6,94) I,J	MIC02690
94	FORMAT(5X,'TEST',3X,I4,' RANKED',15,3X,' IS UNCONNECTED')	MIC02700
	NFLAG=1	MIC02710
	GOTO 383	MIC02720
384	CONTINUE	MIC02730
	PR=TATTR(J,1)	MIC02740
		MIC02750

	ALFA=TATIR(J,2)	MIC02760
C	EDUR(I)=DUR(I)+(ENS(PR,ALFA)-1.)#DUR(I)+XMEAN(PROB,RDUR,5)	MIC02770
	EDUR(I)=DUR(I)+(1.-PR)#XMEAN(PROB,RDUR,2)	MIC02780
	GO TO 383	MIC02790
381	EDUR(I)=DUR(I)	MIC02800
383	CONTINUE	MIC02810
	IF(NFLAG.EQ.1) STOP	MIC02820
C	SET LOWER AND UPPER BOUND OF INTENSITY	MIC02830
	DO 970 I=1,NNODE	MIC02840
	ZL(I)=ZLCW	MIC02850
	ZU(I)=ZUP	MIC02860
	IF(ITYPE(I).EQ.1.OR.ITYPE(I).EQ.2.OR.ITYPE(I).EQ.4) THEN	MIC02870
	ZL(I)=1.0	MIC02880
	ZU(I)=1.0	MIC02890
	ENDIF	MIC02900
970	CONTINUE	MIC02910
C		MIC02920
C	START SIMULATION CYCLIES	MIC02930
C	-----	MIC02940
	DO 1999 ICYCLE=1,NCYCLE	MIC02950
C		MIC02960
	NXTVEC=1	MIC02970
	TIME=0.	MIC02980
	LTIME=0	MIC02990
	CLOCK=0.	MIC03000
	TACCUM=0.	MIC03010
	TIAS=0.	MIC03020
	LAST=0	MIC03030
	ISWTCH=0	MIC03040
	NREC=0	MIC03050
	PFINIS=0.	MIC03060
	NACTIV=0	MIC03070
	NREADY=0	MIC03080
	LENGTH=0	MIC03090
		MIC03100
C	RESET SCHEDULE TABLES	MIC03110
	DO 432 I=1,10000	MIC03120
	TIMVEC(I,1)=0	MIC03130
	TIMVEC(I,2)=0	MIC03140
432	TEVENT(I)=0.	MIC03150
	DO 434 I=1,1000	MIC03160
	DO 434 J=1,11	MIC03170
434	ACTIME(I,J)=0	MIC03180
	DO 1779 L=1,NMILE	MIC03190
1779	TMILE(L)=99999	MIC03200
		MIC03210
C	RESET ACCUMULATION	MIC03220
	DO 920 I=1,NKES	MIC03230
920	ACUMRS(I)=0.	MIC03240
	DO 921 I=1,NSHOP	MIC03250
	TCTPHR(I)=0.	MIC03260
921	ACUMSP(I)=0.	MIC03270
		MIC03280
C	FIND THE ACTUAL DURATION FOR REPAIR ACTIVITIES.	MIC03290
	K=ITURN+1	MIC03300

DO 86 I=K,NNODE	MIC03310
M=I-ITURN	MIC03320
DO 89 N=1,5	MIC03330
89 PRUB(N)=PROBR(M,N)	MIC03340
CALL PICK(PROB,KM,5)	MIC03350
86 DUR(I)=CURR(M,KM)	MIC03360
C RANDOMIZE THE INPUT REQUIREMENTS	MIC03370
DO 1740 I=1,NNODE	MIC03380
NRS=ANUM(I)	MIC03390
IF(NRS.EQ.0) GOTO 1740	MIC03400
IF(ITYPE(I).GT.4) GOTO 1740	MIC03410
DO 1740 J=1,NRS	MIC03420
IF(1SIM.EQ.2) THEN	MIC03430
CALL RAND(NRAND,URAND)	MIC03440
A(I,J)=A(I,J)*(URAND*.4 + .8)	MIC03450
ENDIF	MIC03460
IF(1SIM.EQ.1) A(I,J)=A(I,J)	MIC03470
1740 CONTINUE	MIC03480
C SET RESOURCE CAPACITIES	MIC03490
C	MIC03500
DO 483 L=1,NSHUP	MIC03510
483 CAP(L)=CIC(1,L)	MIC03520
C	MIC03530
C INITIALIZE ACTIVITY DATA	MIC03540
DO 382 I=1,NNODE	MIC03550
TFINIS(I)=99999.	MIC03560
ATFIN(I)=99999.	MIC03570
STATUS(I)=NP(I)	MIC03580
Z(I)=0.	MIC03590
PCV(I)=0.	MIC03600
TLU(I)=0.	MIC03610
382 CONTINUE	MIC03620
C	MIC03630
C START SIMULATION	MIC03640
C	MIC03650
WRITE(6,*) '## START SIMULATION'	MIC03660
WRITE(6,*) '## RUN CYCLE = ',ICYCLE	MIC03670
C	MIC03680
C CALCULATE ES AND LS BY CPM AND SET INITIAL PRIORITIES	MIC03690
C	MIC03700
CALL LSCHED(FDUR,TFINIS,ISCFIN,ITYPE)	MIC03710
CALL SORT(LS,PK1,NNODE,'NWRITE')	MIC03720
C	MIC03730
C SCHEDULE CAPACITY CHANGES	MIC03740
IF(NCC.LE.1) GOTO 313	MIC03750
DO 543 I=2,NCC	MIC03760
TM=TFCC(I)	MIC03770
543 CALL SCHED(7500 + I, TM)	MIC03780
C	MIC03790
C SCHEDULE REPORTING DATES	MIC03800
313 IF(NREPDT.EQ.0) GOTO 314	MIC03810
DO 544 I=1,NREPDT	MIC03820
TM=REPDAT(I)	MIC03830
CALL SCHED(8000 + I, TM)	MIC03840
	MIC03850

544	CONTINUE	MIC03860
314	CONTINUE	MIC03870
C	INITIALIZE AVAILABLE RESOURCES	MIC03880
	DC 132 J=1,NSHOP	MIC03890
132	AP(J)=CAP(J)	MIC03900
C	INITIALIZE INTENSITY MATRIX	MIC03910
	DC 244 I=1,NNODE	MIC03920
	Z(I)=0.	MIC03930
	PCV(I)=0.	MIC03940
244	TLU(I)=0.	MIC03950
C		MIC03960
C	PUT ACTIVITIES WITHOUT PREDECESSOR INTO READY LIST	MIC03970
	NACTIV=0	MIC03980
	NREADY=0	MIC03990
	DC 112 I=1,ITURN	MIC04000
	IF(NP(I).NE.0) GO TO 112	MIC04010
C	IF(ITYPE(I).GT.4) GO TO 2112	MIC04020
	IF(ITYPE(I).GE.3) GO TO 2112	MIC04030
	CALL ADDL(I,PRI,IREADY,NREADY)	MIC04040
	GC TO 112	MIC04050
2112	TSTART(I)=C.	MIC04060
	TFINIS(I)=C.	MIC04070
C	IF(ITYPE(I).EQ.5) TFINIS(I)=DUR(I)	MIC04080
	CALL ADDL(I,PRI,IACTIV,NACTIV)	MIC04090
	CALL SCHED(2500+I,TFINIS(I))	MIC04100
	Z(I)=ZU(I)	MIC04110
112	CONTINUE	MIC04120
C		MIC04130
C	***** START SIMULATION *****	MIC04140
C		MIC04150
	GOTO 810	MIC04160
C	GET THE NEXT EVENT FROM THE LIST	MIC04170
C		MIC04180
41	CALL RMV(CODE,TIME,LAST,CLOCK)	MIC04190
	ELTM=TIME-CLOCK	MIC04200
	CLOCK=TIME	MIC04210
	IF(LAST.EQ.3) WRITE(6,*) 'TIME LIMIT EXCEEDED'	MIC04220
	+ , 'TIME=',CLOCK, ' CODE=',CODE	MIC04230
	IF(LAST.GE.2) GOTO 91	MIC04240
C		MIC04250
C	CCODE= 0+ : START DELAYED FOLLOWER OF A MILESTONE.	MIC04260
C	CCODE=2500+ : AN ACTIVITY HAS FINISHED. START FOLLOWERS.	MIC04270
C	CCODE=5000+ : FLOW-TRANSFERRED FOLLOWER START	MIC04280
C	CCODE=7500+ : CAPACITY IS CHANGED.	MIC04290
C	CCODE=8000+ : REPORTING DATE	MIC04300
C		MIC04310
	IF(CODE.LT.2500) GOTO 51	MIC04320
	IF(CODE.LT.5000) GOTO 61	MIC04330
	IF(CODE.LT.7500) GOTO 71	MIC04340
	IF(CODE.LT.8000) GOTO 222	MIC04350
C		MIC04360
C	REPORTING DATE ... REPORT RESOURCE AVERAGES	MIC04370
C		MIC04380
	WRITE(6,*) 'REPORTING EVENT. TIME IS',CLOCK	MIC04390
	CALL ACCUM(TIAS,WCSHOP,NSHOP)	MIC04400

710	TIAS=CLOCK	MIC04410
	DELT=CLOCK-TACCU	MIC04420
	IF(DELT.EQ.0.) GOTO 713	MIC04430
C		MIC04440
C	CALCULATE AVG. RESOURCES USED BETWEEN DELT	MIC04450
	DO 711 IRES=1,NRES	MIC04460
711	ACCWC(IRES)=ACUMRS(IRES)/DELT	MIC04470
	DO 712 ISHCP=1,NSHOP	MIC04480
	TOTMHR(ISHOP)=TOTMHR(ISHOP)+ACUMSP(ISHOP)	MIC04490
712	ACSHOP(ISHOP)=ACUMSP(ISHOP)/DELT	MIC04500
	TACCU=CLOCK	MIC04510
C		MIC04520
C	WRITE(25,714) (ACCWC(IRES),IRES=1,NRES)	MIC04530
	WRITE(26,714) (ACSHOP(ISHOP),ISHOP=1,NSHOP)	MIC04540
714	FORMAT(1018)	MIC04550
713	CONTINUE	MIC04560
	DO 716 IRES=1,NRES	MIC04570
716	ACUMRS(IRES)=0.	MIC04580
	DO 717 ISHOP=1,NSHOP	MIC04590
717	ACUMSP(ISHOP)=0.	MIC04600
	IF(LAST.EQ.1) GOTO 81	MIC04610
	GOTO 41	MIC04620
C		MIC04630
C	START A DELAYED FOLLOWER OF MILESTONE	MIC04640
51	I=CCDE	MIC04650
	CALL ADDL(1,PRI,IREADY,NREADY)	MIC04660
	IF(LAST.EQ.1) GOTO 81	MIC04670
	GO TO 41	MIC04680
C		MIC04690
C	ACTIVITY IS FINISHED .RELEASE RESOURCES AND UPDATE THE	MIC04700
C	STATUS OF THE FOLLOWERS.	MIC04710
C		MIC04720
61	I=CCDE-2500	MIC04730
	PCOV=1.	MIC04740
	LENGTH=LENGTH+1	MIC04750
		MIC04760
	IF(TIAS.LT.CLOCK) CALL ACCUM(TIAS,WCSHOP,NSHOP)	MIC04770
		MIC04780
C		MIC04790
C	RELEASE RESOURCES	MIC04800
	NRS=ANUM(I)	MIC04810
	IF(NRS.EQ.0) GOTO 252	MIC04820
	DO 201 K=1,NRS	MIC04830
	L=WCSHOP(ASHOP(I,K))	MIC04840
	AR(L)=AR(L)+A(I,K)*Z(I)	MIC04850
201	CONTINUE	MIC04860
252	CONTINUE	MIC04870
C		MIC04880
C	CHECK THE TYPE OF THE ACTIVITY	MIC04890
	IF(IITYPE(I).NE.2) GOTO 212	MIC04900
C	ACTIVITY IS A TEST	MIC04910
C	DRAW A RANDOM NUMBER	MIC04920
	CALL RAND(RKAND,URAND)	MIC04930
	IK=IRANK(I)	MIC04940
	IF(URAND.LE.TATTR(IK,1))GOTO 212	MIC04950
C	FIND THE REPAIR TYPE NEEDED	

CALL RAND(NRAND,URAND)	MIC0496C
SET=TATTR(IK,4)	MIC0497C
IF(SET.LE.0.) GOTO 212	MIC0498C
DO 202 L=1,2	MIC0499C
IF(URAND.LE.SET) GOTO 414	MIC0500C
SET=SET+TATTR(IK,L+4)	MIC0501C
202 CONTINUE	MIC0502C
414 K=TFOLL(IK,L)	MIC0503C
ITYRPR=L	MIC0504C
C UPDATE THE STATUS OF THE FOLLOWER	MIC0505C
STATUS(K)=STATUS(K)-1	MIC0506C
IF(STATUS(K).GT.0) GOTO 241	MIC0507C
C FIND THE TYPE OF THE ACTIVITY	MIC0508C
IF(ITYPE(K).LE.2) GOTO 306	MIC0509C
IF(ITYPE(K).EQ.5) GOTO 506	MIC0510C
IF(ITYPE(K).EQ.9) GOTO 406	MIC0511C
306 CALL ADDL(K,PRI,IREADY,NREADY)	MIC0512C
GOTO 241	MIC0513C
406 TSTART(K)=CLOCK	MIC0514C
TFINIS(K)=CLOCK	MIC0515C
CALL ADDL(K,PRI,IACTIV,NACTIV)	MIC0516C
CALL SCHED(2500+K,CLOCK)	MIC0517C
Z(K)=1.	MIC0518C
GOTO 241	MIC0519C
506 ADUR=DUR(K)+CLOCK	MIC0520C
CALL SCHED(2500+K,ADUR)	MIC0521C
TSTART(K)=CLOCK	MIC0522C
CALL ADDL(K,PRI,IACTIV,NACTIV)	MIC0523C
TFINIS(K)=ADUR	MIC0524C
Z(K)=1.	MIC0525C
241 CONTINUE	MIC0526C
C UPDATE THE ATTRIBUTES OF TEST FOR NEXT OCCURANCE	MIC0527C
TATTR(IK,1)=1.	MIC0528C
DUR(I)=1	MIC0529C
C FIND EXPECTED DURATIONS OF REPAIR AFTER TEST	MIC0530C
DO 584 L=1,2	MIC0531C
PROB(L)=TATTR(IK,3+L)	MIC0532C
K=TFOLL(IK,L)	MIC0533C
RCUR(L)=0.	MIC0534C
IF(K.EQ.0) GOTO 594	MIC0535C
RCUR(L)=DUR(K)	MIC0536C
NBA=TC(I)	MIC0537C
585 IS=FPCS(K)+1	MIC0538C
NBK=TC(K)	MIC0539C
C CHECK IF TC NODE OF FOLLOWER(K) IS EQUAL TO TC NODE OF TEST	MIC0540C
IF(NBA.EQ.NBK)GO TO 584	MIC0541C
RCUR(L)=RCUR(L)+DUR(K)	MIC0542C
K=FOLL(IS)	MIC0543C
GOTO 585	MIC0544C
584 CONTINUE	MIC0545C
EDUR(I)=FDUR(ITYRPR)	MIC0546C
TFINIS(I)=FDUR(ITYRPR)+CLOCK	MIC0547C
GOTO 144	MIC0548C
212 CONTINUE	MIC0549C
C	MIC0550C

C	UPDATE THE STATUS OF THE FOLLOWER	MIC05510
C	NPF=FPOS(I)+1	MIC05520
	KPF=NF(I)+FPOS(I)	MIC05530
	IF(NF(I).EQ.0) GOTO 144	MIC05540
	DO 24 L=NPF,KPF	MIC05550
	IF(CTRAF(L).NE.1) GOTO 24	MIC05560
	K=FCLL(L)	MIC05570
	STATUS(K)=STATUS(K)-1	MIC05580
	IF(STATUS(K).GT.0) GOTO 24	MIC05590
C	RELEASE FOLLOWER	MIC05600
	IF(ITYPE(K).LE.4) GOTO 30	MIC05610
	IF(ITYPE(K).EQ.5) GOTO 50	MIC05620
	IF(ITYPE(K).EQ.7) GOTO 45	MIC05630
	IF(ITYPE(K).EQ.9) GOTO 40	MIC05640
	WRITE(6,*)'ITYPE ERROR FOR ACT',K,'ITYPE(K)=',ITYPE(K)	MIC05650
C		MIC05660
C	FOLLOWER IS A NORMAL ACTIVITY	MIC05670
30	CONTINUE	MIC05680
	IF(ISWTCH.EQ.1) THEN	MIC05690
C	DELAY THE FOLLOWER OF MILESTONE THAT HAS PREDEFINED DATE	MIC05700
	CALL SCHED(K,DMILE)	MIC05710
	GO TO 24	MIC05720
	ENDIF	MIC05730
	CALL ADDL(K,PRI,IREADY,NREADY)	MIC05740
	GOTO 24	MIC05750
C	DUMMY ACTIVITY	MIC05760
40	TSTART(K)=CLOCK	MIC05770
	TFINIS(K)=CLOCK	MIC05780
	CALL ADDL(K,PRI,IACTIV,NACTIV)	MIC05790
	CALL SCHED(2500+K,CLOCK)	MIC05800
	Z(K)=1.	MIC05810
	GOTO 24	MIC05820
C		MIC05830
C	ARRIVED AT A MILESTON-EVENT	MIC05840
45	CONTINUE	MIC05850
	TMILE(MILE(K))=CLOCK	MIC05860
	TSTART(K)=CLOCK	MIC05870
	TFINIS(K)=CLOCK	MIC05880
	DMILE=DUR(K)	MIC05890
	IF(TMILE(MILE(K)).LT.DMILE) ISWTCH=1	MIC05900
	GOTO 24	MIC05910
C		MIC05920
C	"DELAY" ACTIVITY	MIC05930
50	ADUR=DUR(K)+CLOCK	MIC05940
	TSTART(K)=CLOCK	MIC05950
	TFINIS(K)=ADUR	MIC05960
	Z(K)=1.	MIC05970
	CALL SCHED(2500+K,ADUR)	MIC05980
	CALL ADDL(K,PRI,IACTIV,NACTIV)	MIC05990
24	CONTINUE	MIC06000
	ISWTCH=0	MIC06010
144	CONTINUE	MIC06020
C		MIC06030
C	UPDATE ACTIVE LIST	MIC06040
		MIC06050

CALL CANCEL(I,IACTIV,NACTIV)	MIC06060
IF(LAST.EQ.1) GOTO 81	MIC06070
GOTO 41	MIC06080
C	MIC06090
C FLOW-TRANSFERRED FOLLOWER CAN BE STARTED	MIC06100
C	MIC06110
71 I=CCOE-5000	MIC06120
IF(NF(I).EQ.0) GOTO 302	MIC06130
NPF = FPOS(I)+1	MIC06140
KPF = FPOS(I)+NF(I)	MIC06150
DO 301 L=NPF,KPF	MIC06160
K=FCLL(L)	MIC06170
STATUS(K)=STATUS(K)-1	MIC06180
IF(STATUS(K).GT.0) GOTO 301	MIC06190
IF(DUR(K).LE.0.) GOTO 391	MIC06200
CALL ADDL(K,PRI,IREADY,NREADY)	MIC06210
GOTO 301	MIC06220
391 CALL ADDL(K,PRI,IACTIV,NACTIV)	MIC06230
CALL SCHED(2500+K, CLOCK)	MIC06240
TSTART(K)=CLOCK	MIC06250
IFINIS(K)=CLOCK	MIC06260
Z(K)=ZU(K)	MIC06270
301 CONTINUE	MIC06280
302 IF(LAST.EQ.1) GOTO 81	MIC06290
GOTO 41	MIC06300
C	MIC06310
C LAST EVENT AT CLOCK. ACCUMULATE RESOURCES USED	MIC06320
C	MIC06330
81 CALL ACCUM(TIAS,WCSHOP,NSHOP)	MIC06340
LAST=0	MIC06350
C CHECK FOR END OF SIMULATION	MIC06360
C MODIFIED NOT TO WRITE AVG. RES FOR THE LAST PERIOD...	MIC06370
C 810 IF(NREADY.EQ.0.AND.NACTIV.EQ.0) GOTO 710	MIC06380
810 IF(NREADY.EQ.0.AND.NACTIV.EQ.0) GOTO 41	MIC06390
C	MIC06400
881 CONTINUE	MIC06410
IF(NACTIV.EQ.0) GO TO 870	MIC06420
C CHECK THE ASSIGNMENT POLICY	MIC06430
IF(IRAPCL.NE.2) GO TO 870	MIC06440
C	MIC06450
C UPGRADING AND DOWNGRADING	MIC06460
C -----	MIC06470
C	MIC06480
DO 146 I=1,NACTIV	MIC06490
K=IACTIV(I)	MIC06500
IF(ITYPE(K).GT.2) GO TO 146	MIC06510
ZMEX=Z(K)-ZL(K)	MIC06520
IF(ZMEX.LE.0.) GO TO 146	MIC06530
ZNEW=ZL(K)	MIC06540
C ADD UP RELEASED RESOURCES	MIC06550
NPS=ANUM(K)	MIC06560
DO 148 L=1,NRS	MIC06570
J=WCSHOP(ASHOP(K,L))	MIC06580
148 AR(J)=AR(J)+A(K,L)*ZMEX	MIC06590
CALL RESCHD(K,ZNEW,AR)	MIC06600

146	CONTINUE	MIC06610
C		MIC06620
C	INTENSITY UPGRADING	MIC06630
C	-----	MIC06640
870	CONTINUE	MIC06650
	NRUN=1	MIC06660
882	CONTINUE	MIC06670
	M=0	MIC06680
	IF(NACTIV.EQ.0) GOTO 531	MIC06690
C		MIC06700
	DO 151 I=1,NACTIV	MIC06710
	K=IACTIV(I)	MIC06720
	IF(ITYPE(K).GT.4) GOTO 151	MIC06730
	IF(Z(K).EQ.ZU(K)) GOTO 151	MIC06740
	IF(NRUN.EQ.2) GOTO 190	MIC06750
	IF(LS(K).GT.TFINIS(K)) GOTO 151	MIC06760
190	CONTINUE	MIC06770
	M=M+1	MIC06780
	IREV(M)=K	MIC06790
	DRV(M)=LS(K)-TFINIS(K)	MIC06800
151	CONTINUE	MIC06810
531	CONTINUE	MIC06820
	MTUFN=M	MIC06830
	IF(NREADY.EQ.0) GOTO 532	MIC06840
	DO 641 I=1,NREADY	MIC06850
	M=M+1	MIC06860
	J=IREADY(I)	MIC06870
	IREV(M)=J	MIC06880
641	DRV(M)=LS(J)-CLOCK-DUR(J)	MIC06890
532	CONTINUE	MIC06900
	IF(P.EQ.C) GOTO 200	MIC06910
C		MIC06920
C	SET PRIORITIES ACCORDING TO DRV(LATENESS SCORE)	MIC06930
	CALL SORT(CFV,NPRI,M,'NWRITE')	MIC06940
C		MIC06950
	DO 161 I=1,M	MIC06960
C		MIC06970
	J1=NPRI(I)	MIC06980
	JK=IREV(J1)	MIC06990
	ZMEX=ZU(JK)-Z(JK)	MIC07000
	IF(DUR(JK).LE.0) GOTO 559	MIC07010
	PCOV=PCV(JK)+(CLOCK-TLU(JK))*Z(JK)/DUR(JK)	MIC07020
559	IF(DUR(JK).LE.0) PCOV=1.	MIC07030
	IF(NRUN.EQ.2) GOTO 571	MIC07040
	IF(LS(JK).LE.CLOCK) GOTO 571	MIC07050
	ZC=((1.-PCOV)*DUR(JK))/(LS(JK)-CLOCK))- Z(JK)	MIC07060
	IF(ZMEX.GT.ZC) ZMEX=ZC	MIC07070
571	CONTINUE	MIC07080
C		MIC07090
	NRS=ANUM(JK)	MIC07100
	IF(NRS.EQ.0) GOTO 96	MIC07110
	DO 171 L=1,NRS	MIC07120
	J=WCSHOP(ASHOP(JK,L))	MIC07130
	IF(A(JK,L).LT..0001) GOTO 171	MIC07140
	ZAK=AR(J)/A(JK,L)	MIC07150

	IF(ZMEX.GT.ZAR) ZMEX=ZAR	MIC07160
171	CONTINUE	MIC07170
96	CONTINUE	MIC07180
C		MIC07190
	IF(ZMEX.LE.0.01)GOTO 161	MIC07200
	ZNEW=ZMEX+Z(JK)	MIC07210
C		MIC07220
C	CHECK PREDECESSORS	MIC07230
	IF(NP(JK).EQ.0) GOTO 310	MIC07240
	ISTART=PPDS(JK)+1	MIC07250
	IEND=NP(JK)+PPDS(JK)	MIC07260
C		MIC07270
C	CHECK FLOW TRANSFERS	MIC07280
	DO 210 IPOS=ISTART,IEND	MIC07290
	IF(CTRANP(IPOS).EQ.1) GOTO 210	MIC07300
	IK=PRED(IPOS)	MIC07310
	IF(TFINIS(IK).LE.CLOCK) GOTO 210	MIC07320
	IF(DUR(JK).LE.0) GOTO 210	MIC07330
	ZMAXI=DUR(JK)/(1.-CTRANP(IPOS)-PCOV)/(TFINIS(IK)-CLOCK)	MIC07340
	IF(ZMAXI.LT.ZNEW) ZNEW=ZMAXI	MIC07350
	ZMEX=ZNEW-Z(JK)	MIC07360
210	CONTINUE	MIC07370
310	CONTINUE	MIC07380
C		MIC07390
	IF(ZNEW.LT.ZL(JK)) GOTO 161	MIC07400
C	UPDATE AVAILABLE RESOURCES	MIC07410
	IF(NRS.EQ.0) GOTO 196	MIC07420
	DO 172 K=1,NRS	MIC07430
	L=WCSHOP(ASHOP(JK,K))	MIC07440
	AR(L)=AR(L)-A(JK,K)*ZMEX	MIC07450
172	CONTINUE	MIC07460
196	CONTINUE	MIC07470
C		MIC07480
C	RESCHEDULE FINISH AND PROGRESS	MIC07490
C		MIC07500
	IF(JI.LE.MTURN) GOTO 115	MIC07510
C	FOR NEWLY SCHEDULED ACTIVITIES	MIC07520
	CALL CANCEL(JK,IREADY,NREADY)	MIC07530
	CALL ADDL(JK,PRI,IACTIV,NACTIV)	MIC07540
	TSTART(JK)=CLOCK	MIC07550
	TLU(JK)=CLOCK	MIC07560
115	CONTINUE	MIC07570
C		MIC07580
	IF(ZNEW.EQ.Z(JK)) GOTO 161	MIC07590
	CALL RESCHD(JK,ZNEW,AR)	MIC07600
161	CONTINUE	MIC07610
C		MIC07620
200	CONTINUE	MIC07630
	IF (NRUN.EC.2) GOTO 41	MIC07640
C		MIC07650
C	RESCHEDULE TRULY SLACK ACT'S IF CRITICAL ACT. DELAYED	MIC07660
C		MIC07670
	CALL CRITIC(ZU,PRI)	MIC07680
C		MIC07690
	NRUN=2	MIC07700

	GOTO 892	MIC07710
C		MIC07720
C	RESOURCE CAPACITY CHANGE	MIC07730
C	-----	MIC07740
	222 CONTINUE	MIC07750
	K=CODE-7500	MIC07760
	DO 1121 I=1,NSHOP	MIC07770
	AR(I)=AR(I)+CIC(K,I)-CAP(I)	MIC07780
	CAP(I)=CIC(K,I)	MIC07790
	1121 CONTINUE	MIC07800
C		MIC07810
C	SEARCH IF THERE IS ANY CAPACITY VIOLATION	MIC07820
	DO 1131 I=1,NSHOP	MIC07830
	IF(AR(I).LT.-0.01) GOTO 1141	MIC07840
	1131 CONTINUE	MIC07850
	1231 CONTINUE	MIC07860
	IF(LAST.EQ.1) GOTO 81	MIC07870
	GOTO 41	MIC07880
C		MIC07890
C	DOWNGRADING INTENSITIES	MIC07900
C		MIC07910
C	CREATE LIST OF ACTIVITIES TO DOWNGRADE	MIC07920
	1141 CONTINUE	MIC07930
	WRITE(6,*)'*** INTENSITY DOWNGRADING TRIED ***'	MIC07940
	M=0	MIC07950
	DO 1151 I=1,NACTIV	MIC07960
	K=IACTIV(I)	MIC07970
	IF(ITYPE(K).GT.4) GOTO 1151	MIC07980
	IF(Z(K).EQ.ZL(K)) GOTO 1151	MIC07990
	M=M+1	MIC08000
	IREV(M)=K	MIC08010
	DRV(M)=TFINIS(K)-LS(K)	MIC08020
	1151 CONTINUE	MIC08030
	IF(M.LE.0) GOTO 37	MIC08040
	CALL SORT(DRV,NPRI,M,'NWRITE')	MIC08050
	ZLOW=1.	MIC08060
	IIRUN=1	MIC08070
	CALL DWNGRD(M,ZLOW,Z,IIRUN,PRI)	MIC08080
C		MIC08090
	IF(JFLAG.EQ.0) GOTO 1231	MIC08100
C		MIC08110
C	INTERUPT ACTIVITIES	MIC08120
	37 CONTINUE	MIC08130
	M=0	MIC08140
	DO 2152 I=1,NACTIV	MIC08150
	K=IACTIV(I)	MIC08160
	IF(ITYPE(K).GT.4) GOTO 2152	MIC08170
	IF(Z(K).LE..0001) GOTO 2152	MIC08180
	M=M+1	MIC08190
	IREV(M)=K	MIC08200
	DRV(M)=TFINIS(K)-LS(K)	MIC08210
	2152 CONTINUE	MIC08220
	IF(M.EQ.0) GOTO 1173	MIC08230
	CALL SORT(DRV,NPRI,M,'NWRITE')	MIC08240
	ZLOW=0.	MIC08250

IIRUN=2	MIC0826
CALL DNGRC(M,2LCW,2,IIRUN,PRI)	MIC0827
GO TO 1231	MIC0828
C	MIC0829
1173 CONTINUE	MIC0830
C	MIC0831
THERE IS NO ACTIVITY AVAILABLE	MIC0832
WRITE(6,*) 'NO ACT AVAILABLE IN DOWNGRADING...LOGICAL ERKOK..'	MIC0833
STOP	MIC0834
C	MIC0835
END OF SIMULATION	MIC0836
C	MIC0837
91 WRITE(6,*) 'END OF SIMULATION. TIME IS ',CLOCK	MIC0838
C	MIC0839
WRITE OUTPUT FILES	MIC0840
IF(NMILE.EQ.0) GO TO 95	MIC0841
DO 99 I=1,NMILE	MIC0842
II=TMILE(I)-1	MIC0843
IF(II.EQ.-1) II=0	MIC0844
CALL DCONV(II,IMILE)	MIC0845
SMILE(I)=IMILE(1)*10000+IMILE(2)*100+IMILE(3)	MIC0846
99 CONTINUE	MIC0847
WRITE(57,714) (SMILE(I),I=1,NMILE)	MIC0848
WRITE(27,714) (TMILE(I),I=1,NMILE)	MIC0849
C	MIC0850
95 NSHOPP=NSHOP+1	MIC0851
ATOT(NSHOPP)=0.	MIC0852
DO 5555 KKL=1,NSHOP	MIC0853
ATOT(NSHOPP)=ATOT(NSHOPP)+TOTMHR(KKL)	MIC0854
5555 TOTMDY(KKL)=TOTMHR(KKL)	MIC0855
TOTMDY(NSHOPP)=ATOT(NSHOPP)	MIC0856
WRITE(28,714) (TOTMDY(I),I=1,NSHOPP)	MIC0857
C	MIC0858
IF(ICYCLE.GT.1) GO TO 2002	MIC0859
IF(IRAPCL.EQ.1) WRITE(36,2003) NAME,NCYCLE,ILOW,ILP	MIC0860
IF(IRAPCL.EQ.2) WRITE(36,2004) NAME,NCYCLE,ILOW,ILP	MIC0861
WRITE(36,2005) SMDN,SDAY,SYR	MIC0862
WRITE(36,2006)	MIC0863
C	MIC0864
IF(ICALD.EQ.1) THEN	MIC0865
DO 2000 I=1,NNODE	MIC0866
KSTART=TSTART(I)+.5	MIC0867
KFINIS=TFINIS(I)+.5-1	MIC0868
IF(KFINIS.LT.KSTART) KFINIS=KSTART	MIC0869
IF(ITYPE(I).EQ.7) THEN	MIC0870
II=MILE(I)	MIC0871
KSTART=TMILE(II)-1	MIC0872
IF(KSTART.EQ.-1) KSTART=0	MIC0873
KFINIS=KSTART	MIC0874
ENDIF	MIC0875
IF(ITYPE(I).EQ.9) KSTART=KFINIS	MIC0876
CALL DCONV(KSTART,CALND1)	MIC0877
CALL DCONV(KFINIS,CALND2)	MIC0878
WRITE(36,2001) I,FROM(I),TO(I),CALND1,CALND2	MIC0879
2000 CONTINUE	MIC0880
ELSE	

```

      DO 2009 I=1,NNODE
2009      WRITE(36,2008) I,FROM(I),TO(I),TSTART(I),TFINIS(I)
      ENDIF
2002 CONTINUE
C
C      WRITE RUN-STATUS REPORT
      IF(ICYCLE.EQ.1) THEN
        WRITE(40,1001) NAME
1001      FCRMAT(A50)
        IF(IRAPOL.EQ.1) WRITE(40,1002)
        IF(IRAPOL.EQ.2) WRITE(40,1003)
        IF(NCYCLE.EQ.1) WRITE(40,1004)
        IF(NCYCLE.GT.1) WRITE(40,1005) NCYCLE
        IF(NTEST.EQ.0) WRITE(40,1006)
        IF(NTEST.EQ.1) WRITE(40,1007)
        WRITE(40,1009) NNODE,ITURN,NRES,NSHOP,ILOW,IUP,KTEND,NCC,NKEPDT,
+      ISCFIN,NPILE
      ENDIF
1002 FCRMAT('INTENSITY ASSIGNMENT POLICY : UPGRADING ONLY')
1003 FCRMAT('INTENSITY ASSIGNMENT POLICY : UP & DOWNGRADING')
1004 FCRMAT('NO SIMULATIONS : 1 CYCLE')
1005 FCRMAT('SIMULATIONS : ',15,' CYCLES')
1006 FCRMAT('TEST ACTIVITIES & REWORKS : NO')
1007 FCRMAT('TEST ACTIVITIES & REWORKS : YES')
1009 FCRMAT('NUMBER OF ACTIVITIES : ',15,/
+      'NUM. OF ACT. (NO REWORKS) : ',15,/
+      'NUMBER OF WORKCENTERS : ',15,/
+      'NUMBER OF SHOPS : ',15,/
+      'LOWER BOUND OF INTENSITY : ',15,/
+      'UPPER BOUND OF INTENSITY : ',15,/
+      'LIMIT OF PROJECT FINISH : ',15,' (FOR SIMULATIONS)',/
+      'NUMBER OF CAPACITY CHANGE : ',15,/
+      'NUMBER OF REPORTING DATES : ',15,/
+      'PROJECT TARGET FINISH TIME : ',15,/
+      'NUMBER OF MILESTONES : ',15 )
C
      REWIND 17
      WRITE(17,1777) NRAND
C
C
2003 FCRMAT(1H1,120(1H=),//,6X,'PROJECT NAME : ',A20,15,' RUN(S)',
+      ' USING UPGRADING ONLY',
+      ' INTENSITY RANGE ',15,' - ',15, '//,120('='))
2004 FCRMAT(1H1,120(1H=),//,6X,'PROJECT NAME : ',A20,15,' RUN(S)',
+      ' USING UP & DOWN GRADING',
+      ' INTENSITY RANGE ',15,' - ',15, '//,120('='))
2005 FCRMAT(1H , ' PROJECT START DATE : ',315,/)
2006 FCRMAT(' ACTIVITY',20X,' SCHEDULE',/,
+      ' INTERNAL #',3X,' I',5X,' J',4X,
+      ' START DATE',11X,' FINISH DATE',/,70('--'))
2001 FCRMAT(5X,14,5X,A4,5X,A4,5X,3(1X,12),13X,3(1X,12))
2006 FCRMAT(5X,14,5X,A4,5X,A4,7X,F7.1,15X,F7.1)
323 FCRMAT(A50,8(/49X,16))
1777 FCRMAT(110)

```

MIC08810
 MIC08820
 MIC08830
 MIC08840
 MIC08850
 MIC08860
 MIC08870
 MIC08880
 MIC08890
 MIC08900
 MIC08910
 MIC08920
 MIC08930
 MIC08940
 MIC08950
 MIC08960
 MIC08970
 MIC08980
 MIC08990
 MIC09000
 MIC09010
 MIC09020
 MIC09030
 MIC09040
 MIC09050
 MIC09060
 MIC09070
 MIC09080
 MIC09090
 MIC09100
 MIC09110
 MIC09120
 MIC09130
 MIC09140
 MIC09150
 MIC09160
 MIC09170
 MIC09180
 MIC09190
 MIC09200
 MIC09210
 MIC09220
 MIC09230
 MIC09240
 MIC09250
 MIC09260
 MIC09270
 MIC09280
 MIC09290
 MIC09300
 MIC09310
 MIC09320
 MIC09330
 MIC09340
 MIC09350

951	FORMAT(20I4)	MIC0936C
953	FORMAT(20F4.2)	MIC0937C
C 954	FORMAT(4F10.1)	MIC0938C
952	FORMAT(6X,A4,2X,A4,3X,I4,2X,I1)	MIC0939C
C 952	FORMAT(6X,A4,9X,I4,2X,I1)	MIC0940C
961	FORMAT(5X,4I5)	MIC0941C
963	FORMAT(10F8.2)	MIC0942C
30C1	FORMAT(10F8.2)	MIC0943C
1999	CONTINUE	MIC0944C
9999	STOP	MIC0945C
	END	MIC0946C
		MIC0947C
		MIC0948C
		MIC0949C
	SUBROUTINE CANCEL(I,IX,LX)	MIC0950C
C	THIS SUBROUTINE CANCELS THE INDEX(I) FROM THE LIST (IX)	MIC0951C
C	AND UPDATES THE LIST LENGTH (LX)	MIC0952C
	DIMENSION IX(1000)	MIC0953C
	IF(LX.EQ.0) GOTO 30	MIC0954C
	NA=1	MIC0955C
	DO 12 K=1,LX	MIC0956C
	IF(NA.EQ.IX(K)) GO TO 22	MIC0957C
12	CONTINUE	MIC0958C
C	CODE "I" DOES NOT EXIT IN THE LIST.	MIC0959C
	RETURN	MIC0960C
22	LX=LX-1	MIC0961C
	N=K	MIC0962C
	DO 14 J=N,LX	MIC0963C
	IX(J)=IX(J+1)	MIC0964C
14	CONTINUE	MIC0965C
	IX(LX+1)=0	MIC0966C
30	CONTINUE	MIC0967C
	RETURN	MIC0968C
	END	MIC0969C
		MIC0970C
		MIC0971C
		MIC0972C
	SUBROUTINE ADDL(I,PRI,IX,LX)	MIC0973C
C	THIS SUBROUTINE ADDS INDEX (I) TO THE LIST (IX) AND UPDATES LIST	MIC0974C
C	LENGTH, (LX)	MIC0975C
	DIMENSION IX(1000), IY(1000)	MIC0976C
	INTEGER PRI(2500)	MIC0977C
	NA=1	MIC0978C
	IF(LX.EQ.0) GOTO 30	MIC0979C
	DO 12 K=1,LX	MIC0980C
	JA=IX(K)	MIC0981C
	IF(PRI(NA).LT.PRI(JA)) GOTO 20	MIC0982C
12	CONTINUE	MIC0983C
	LX=LX+1	MIC0984C
	IX(LX)=NA	MIC0985C
	GOTO 40	MIC0986C
20	DO 14 J=1,LX	MIC0987C
14	IY(J+1)=IX(J)	MIC0988C
	IX(K)=NA	MIC0989C
		MIC0990C

```

      LX=LX+1
      M=M+1
      DO 16 J=M,LX
16    IX(J)=IY(J)
      GOTC 40
30    IX(1)=NA
      LX=1
40    RETURN
      END

```

```

      SUBROUTINE SORT(B1,P1,KLIMIT,WRTCOD)

```

```

C      THIS SUBROUTINE SORTS A VECTOR B1, WHILE RESERVING ITS
C      ORIGINAL ORDER IN VECTOR P1. THE SUBROUTINE USES THE
C      "QUICKSORT" METHOD.

```

```

      REAL B(4500),VECCUT(4500),B1(KLIMIT)
      INTEGER P(4500),PCUT(4500),P1(KLIMIT)
      CHARACTER*7 WRTCOD
      DO 100 I=1,KLIMIT
      B(I)=B1(I)
100    P(I)=I
      ALIMIT=FLUAT(KLIMIT)
      ANUM=ALCG10(ALIMIT)/ALOG10(2.)
      NUM=ANUM
      IF(ANUM.GT.NUM) NUM =NUM +1
      MM=2*NUM
      NRESID=MM-KLIMIT
      DO 101 L=1,NRESID
101    B(KLIMIT+L)=1.E10
      DO 6 I=1,NUM
      K=2*1
      M=MP/K
      DO 4 J1=1,M
      LSTART=K*(J1-1)+1
      LEND= LSTART+K-1
      K2=K/2
      K1=0
      J2=LSTART
1    IF(B(J2).LE.B(J2+K2)) GOTO 2
      VECCUT(LSTART+K1)=B(J2+K2)
      PCUT(LSTART+K1)=P(J2+K2)
      B(J2+K2)=1.E10
      K2=K2+1
      GOTO 3
2    PCUT(LSTART+K1)=P(J2)
      VECCUT(LSTART+K1)=B(J2)
      B(J2)=1.E10
      J2=J2+1
      K2=K2-1
3    J3=J2-LSTART+1
      IF(J3.GT.(K/2)) GOTO 31
      IF((J3+K2).GT.K) GOTO 31
      K1=K1+1

```

```

MIC09910
MIC09920
MIC09930
MIC09940
MIC09950
MIC09960
MIC09970
MIC09980
MIC09990
MIC10000
MIC10010
MIC10020
MIC10030
MIC10040
MIC10050
MIC10060
MIC10070
MIC10080
MIC10090
MIC10100
MIC10110
MIC10120
MIC10130
MIC10140
MIC10150
MIC10160
MIC10170
MIC10180
MIC10190
MIC10200
MIC10210
MIC10220
MIC10230
MIC10240
MIC10250
MIC10260
MIC10270
MIC10280
MIC10290
MIC10300
MIC10310
MIC10320
MIC10330
MIC10340
MIC10350
MIC10360
MIC10370
MIC10380
MIC10390
MIC10400
MIC10410
MIC10420
MIC10430
MIC10440
MIC10450

```

```

      GOTO 1
31  DC 32 L=LSTART,LEND
      IF(B(L).GE.1.E10) GOTO 32
      K1=K1+1
      VECOUT(LSTART+K1)=B(L)
      POUT(LSTART+K1)=P(L)
32  CONTINUE
      4  CONTINUE
      DO 5 J=1,KLIMIT
        P(J)=PLUT(J)
      5  B(J)=VECCUT(J)
      6  CONTINUE
      DO 7 I=1,KLIMIT
        IF(WRTCOD.NE.'NOWRITE' ) B1(I)=B(I)
      7  P1(P(I))=I
      RETURN
      END

      SUBROUTINE PICK(P,K,KLIM)
      DIMENSION P(100)
      SET=0.
      DO 20 I=1,KLIM
        SET=SET+P(I)
        CALL RAND(NRAND,URAND)
        IF(URAND.LE.SET) GOTO 30
20  CONTINUE
30  K=I
      RETURN
      END

      FUNCTION ENS(PS,ALFA)
      THIS FUNCTION CALCULATES EXPECTED NUMBER OF TRIALS NEEDED
      C A TEST ACTIVITY TO BE SUCCESSFUL, PS=PROBABILITY OF SUCCESS
      C AT THE CURRENT TRIAL AND ALFA CONSTANT INCREASE IN PROBABILITY
      C OF SUCCESS AFTER EACH FAILURE.
      ENS=0.
      PROS=PS
      PROF=1.
      DO 10 I=1,20
        AD=PROS*PROF*I
        ENS =ENS+AD
        IF(AD.LE.0.001) GOTO 20
        PROF=PROF*(1.-PROS)
        PROS=PROS*ALFA
10  CONTINUE
20  RETURN
      END

      REAL FUNCTION XMEAN(PROB,VAL,NUM)
      DIMENSION PROB(10), VAL(10)
      C THIS FUNCTION CALCULATES MEAN OF AN ARRAY(VAL), WITH
      C PROBABILITIES (PROB), NUM BEING THE NUMBER OF ELEMENTS
      C IN VAL

```

MIC1046C
 MIC1047C
 MIC1048C
 MIC1049C
 MIC1050C
 MIC1051C
 MIC1052C
 MIC1053C
 MIC1054C
 MIC1055C
 MIC1056C
 MIC1057C
 MIC1058C
 MIC1059C
 MIC1060C
 MIC1061C
 MIC1062C
 MIC1063C
 MIC1064C
 MIC1065C
 MIC1066C
 MIC1067C
 MIC1068C
 MIC1069C
 MIC1070C
 MIC1071C
 MIC1072C
 MIC1073C
 MIC1074C
 MIC1075C
 MIC1076C
 MIC1077C
 MIC1078C
 MIC1079C
 MIC1080C
 MIC1081C
 MIC1082C
 MIC1083C
 MIC1084C
 MIC1085C
 MIC1086C
 MIC1087C
 MIC1088C
 MIC1089C
 MIC1090C
 MIC1091C
 MIC1092C
 MIC1093C
 MIC1094C
 MIC1095C
 MIC1096C
 MIC1097C
 MIC1098C
 MIC1099C
 MIC1100C

```

XMEAN=0.
DC 10 I=1,NUM
10 XMEAN=XMEAN+PROB(I)*VAL(I)
RETURN
END

```

```

SUBROUTINE ADJFIN(EDUR,ATFIN,NNODE)
DIMENSION PCV(2500), TLU(2500), TFINIS(2500),Z(2500)
DIMENSION ATFIN(2500),EDUR(2500)
INTEGER DUR(2500)
COMMON/BBB/ PCV,TLU,TFINIS,Z,CLOCK,DUR,LENGTH

```

C THIS SUBROUTINE FINDS ADJUSTED FINISH TIMES FOR TARGET FINISH
C CALCULATIONS.

```

DC 10 I=1,NNODE
IF(TFINIS(I).EQ.99999.) GOTO 11
IF(TFINIS(I).LE.CLOCK) GOTO 11
PCOV=PCV(I)+(CLOCK-TLU(I))*Z(I)/DUR(I)
IF(Z(I).LE.0. .AND. PCOV.LE.0.) GOTO 11
ATFIN(I)=CLOCK+EDUR(I)-PCOV*DUR(I)
GOTO 10
11 ATFIN(I)=TFINIS(I)
10 CONTINUE
RETURN
END

```

```

SUBROUTINE LSCHED(DUR,TFINIS,ISCFIN,ITYPE)

```

C THIS SUBROUTINE FINDS EARLY & LATE FINISH TIMES FOR ACTIVITIES
C BY CPM

```

INTEGER RANK(2500),ITYPE(2500)
DIMENSION EF(2500), DUR(2500)
REAL LS(2500),LFNA,TFINIS(2500)
INTEGER FOLL(10000),NF(2500),FPOS(2500)
INTEGER PRED(10000),NP(2500),PPOS(2500)
DIMENSION CTRANF(10000),CTRANP(10000)
COMMON/BB4/CTRANF,CTRANP
COMMON/BB5/ LS,NNODE, RANK, ITURN
COMMON/BB6/PRED, NP, PPOS
COMMON/BB7/FOLL, NF, FPOS

```

C FORWARD ROUTINE

```

FINIS=0.
I=RANK(1)
IF(TFINIS(I).NE.99999.) GOTO 50
EF(I)=DUR(I)
GOTO 60
50 EF(I)=TFINIS(I)
60 CONTINUE
DC 100 K=2,ITURN
NA=RANK(K)
IF(TFINIS(NA).EQ.99999. ) GOTO 70
EF(NA)=TFINIS(NA)
GOTO 71

```

MIC11010
MIC11020
MIC11030
MIC11040
MIC11050
MIC11060
MIC11070
MIC11080
MIC11090
MIC11100
MIC11110
MIC11120
MIC11130
MIC11140
MIC11150
MIC11160
MIC11170
MIC11180
MIC11190
MIC11200
MIC11210
MIC11220
MIC11230
MIC11240
MIC11250
MIC11260
MIC11270
MIC11280
MIC11290
MIC11300
MIC11310
MIC11320
MIC11330
MIC11340
MIC11350
MIC11360
MIC11370
MIC11380
MIC11390
MIC11400
MIC11410
MIC11420
MIC11430
MIC11440
MIC11450
MIC11460
MIC11470
MIC11480
MIC11490
MIC11500
MIC11510
MIC11520
MIC11530
MIC11540
MIC11550

70	CONTINUE	MIC1156
	ESNA=0.	MIC1157
	IF(NP(NA).EQ.0) GOTO 201	MIC1158
	JP1=PPOS(NA)+1	MIC1159
	JP2=PPOS(NA)+NP(NA)	MIC1160
	DO 200 J=JP1,JP2	MIC1161
	EFDUR=DUR(NA)	MIC1162
	JA=PRED(J)	MIC1163
	IF(EFDUR.GT.DUR(JA)) EFDUR=DUR(JA)	MIC1164
	EFC=EF(JA)-(1.-CTRANP(J))*EFDUR	MIC1165
	ESNA=AMAX1(ESNA,EFC)	MIC1166
200	CONTINUE	MIC1167
201	CONTINUE	MIC1168
	EF(NA)=ESNA+DUR(NA)	MIC1169
	IF(ITYPE(NA).EQ.7) EF(NA)=ESNA	MIC1170
71	CONTINUE	MIC1171
	FINIS=AMAX1(FINIS,EF(NA))	MIC1172
100	CONTINUE	MIC1173
C	WRITE(24,12) FINIS	MIC1174
C	WRITE(24,494) (I,EF(I),I=1,NNODE)	MIC1175
C	12 FORMAT(' PROJECT FINISH TIME = ',F8.2)	MIC1176
C		MIC1177
C	BACKWARD ROUTINE	MIC1178
C		MIC1179
	IF(ISCFIN.NE.0.AND.ISCFIN.LT.FINIS) THEN	MIC1180
	WRITE(6,*) '***** ERROR *****'	MIC1181
	WRITE(6,*) 'SCHEDULE TARGET FINISH IS LESS THAN FORWARD FINISH'	MIC1182
	WRITE(6,*) 'TARGET = ',ISCFIN, ' FINIS = ',FINIS	MIC1183
	STOP	MIC1184
	ENDIF	MIC1185
	IF(ISCFIN.GT.0) FINIS=ISCFIN	MIC1186
C		MIC1187
	NA=FANK(ITURN)	MIC1188
	LS(NA)=FINIS-DUR(NA)	MIC1189
	IF(ITYPE(NA).EQ.7) THEN	MIC1190
	LS(NA)=FINIS	MIC1191
	IF(LS(NA).GT.DUR(NA).AND.EF(NA).LT.DUR(NA)) LS(NA)=DUR(NA)	MIC1192
	IF(LS(NA).LT.DUR(NA)) THEN	MIC1193
	WRITE(6,*) ' ** MILESTONE DATE FOR NODE',NA	MIC1194
	WRITE(6,*) ' IS INACTIVE. PROGRAM WILL USE MORE'	MIC1195
	WRITE(6,*) ' BINDING DATE AND CONTINUE.'	MIC1196
	DUR(NA)=0	MIC1197
	ENDIF	MIC1198
	ENDIF	MIC1199
	K=ITURN-1	MIC1200
	DO 300 JJ=1,K	MIC1201
	KK=ITURN-JJ	MIC1202
	NA=FANK(KK)	MIC1203
	LFNA=FINIS	MIC1204
	IF(NF(NA).EQ.0) GOTO 401	MIC1205
	JF1=FPOS(NA)+1	MIC1206
	JF2=FPOS(NA)+NF(NA)	MIC1207
	DO 400 J=JF1,JF2	MIC1208
	EFULR=DUR(NA)	MIC1209
	JA=FOLL(J)	MIC1210

```

      IF(IJTYPE(JA).EQ.7) EFDUR=0
      IF(EFDUR.GT.DUR(JA)) EFDUR=DUR(JA)
      LSC=LS(JA)+(1.-CTRANF(J))#EFDUR
      IF(LFNA.GE.LSC) LFNA=LSC
400  CONTINUE
401  CONTINUE
      LS(NA)=LFNA-DUR(NA)
C    SET MILESTONE DATE
      IF(IJTYPE(NA).EQ.7) THEN
        LS(NA)=LFNA
        IF(LS(NA).GT.DUR(NA).AND.EF(NA).LT.DUR(NA)) LS(NA)=DUR(NA)
        IF(LS(NA).LT.DUR(NA)) THEN
          WRITE(6,*) ' ** MILESTONE DATE FOR NODE',NA
          WRITE(6,*) ' IS INACTIVE. PROGRAM WILL USE MORE'
          WRITE(6,*) ' BINDING DATE AND CONTINUE.'
          DUR(NA)=0
        ENDIF
      ENDIF
300  CONTINUE
      DO 80 I=1,NNODE
      IF(I.LE.ITURN) GOTO 90
      LS(I)=DUR(I)
      GOTO 80
90   LS(I)=LS(I)+DUR(I)
      IF(IJTYPE(I).EQ.7) LS(I)=LS(I)-DUR(I)
80   CONTINUE
C    WRITE(24,494) (I,LS(I),I=1,NNODE)
C 494  FORMAT(5(I8,F8.2))

      RETURN
      END

      SUBROUTINE CHECK(XAF,IX)
C
C    THIS SUBROUTINE CHECKS IF CAPACITIES ARE VIOLATED.
      DIMENSION XAR(200)
      INTEGER WCSHOP(200),NSHOP
      COMMON/CC9/WCSHOP,NSHOP
      IX=0
      DO 13 I=1,NSHOP
      IF(XAR(I).LT.-.001) IX=1
13   CONTINUE
      RETURN
      END

      SUBROUTINE CRITIC(ZU,PRI)
C
C    THIS ROUTINE DELAYES "TRULY SLACK" ACTIVITIES TL GET RESOURCES
C    ENOUGH TO START CRITICAL ACTIVITY THAT HAS DELAYED
C
      DIMENSION A(2500,70),AR(200),KNEED(200)
      INTEGER ASHOP(2500,70),ANUM(2500)
      DIMENSION ZL(2500),Z(2500),ZU(2500)
      DIMENSION ISTART(2500),TFINIS(2500)

```

MIC12110
 MIC12120
 MIC12130
 MIC12140
 MIC12150
 MIC12160
 MIC12170
 MIC12180
 MIC12190
 MIC12200
 MIC12210
 MIC12220
 MIC12230
 MIC12240
 MIC12250
 MIC12260
 MIC12270
 MIC12280
 MIC12290
 MIC12300
 MIC12310
 MIC12320
 MIC12330
 MIC12340
 MIC12350
 MIC12360
 MIC12370
 MIC12380
 MIC12390
 MIC12400
 MIC12410
 MIC12420
 MIC12430
 MIC12440
 MIC12450
 MIC12460
 MIC12470
 MIC12480
 MIC12490
 MIC12500
 MIC12510
 MIC12520
 MIC12530
 MIC12540
 MIC12550
 MIC12560
 MIC12570
 MIC12580
 MIC12590
 MIC12600
 MIC12610
 MIC12620
 MIC12630
 MIC12640
 MIC12650

	REAL LS(2500),PCV(2500),TLU(2500)	MIC1266C
	INTEGER WCSHOP(200),NSHOP	MIC1267C
	INTEGER PRED(10000),NP(2500),PPOS(2500)	MIC1268C
	REAL CTRANF(10000),CTRANP(10000)	MIC1269C
	INTEGER PRI(2500),DUR(2500),RANK(2500)	MIC1270C
	INTEGER IACTIV(1000),IREADY(1000)	MIC1271C
	INTEGER NSAVE(900),MSAVE(900),IRESCH(900)	MIC1272C
	COMMON/BB1/ IREADY, IACTIV,NREADY,NACTIV	MIC1273C
	COMMON/BB4/CTRANF,CTRANP	MIC1274C
	COMMON/BB5/ LS,NNODE, RANK, ITURN	MIC1275C
	COMMON/BB6/PRED, NP, PPOS	MIC1276C
	COMMON/BB8/PCV,TLU,TFINIS,Z,CLOCK,DUR,LENGTH	MIC1277C
	COMMON/BB9/ANUM, ASHOP, A, AR	MIC1278C
	COMMON/BB11/ZL	MIC1279C
	COMMON/BB13/NRES, NCC	MIC1280C
	COMMON/CC9/ WCSHOP, NSHOP	MIC1281C
C	ISAVE=0	MIC1282C
	JSAVE=0	MIC1283C
C		MIC1284C
	DO 10 II=1,NREADY	MIC1285C
C		MIC1286C
C	SEARCH CRITICAL ACTIVITY	MIC1287C
	I=IREADY(II)	MIC1288C
	IF((LS(I)-CLOCK-DUR(I)).GT.0) GO TO 1C	MIC1289C
	MRES=ANUM(I)	MIC1290C
	IF(MRES.EQ.0) GO TO 10	MIC1291C
C		MIC1292C
	WRITE(37,100) CLOCK,I	MIC1293C
100	FORMAT(1X,F6.2,' ACT.',14,' IN CRITICAL ROUTINE *****')	MIC1294C
C		MIC1295C
C	SET RESOURCES NEEDED	MIC1296C
	DO 12 L=1,MRES	MIC1297C
	K=WCSHOP(ASHOP(I,L))	MIC1298C
12	RNEED(K)=A(I,L)*ZL(I)-AR(K)	MIC1299C
C		MIC1300C
	DO 13 L=1,900	MIC1301C
13	IRESCH(L)=0	MIC1302C
C		MIC1303C
C	SEARCH TRULY SLACK ACTIVITIES	MIC1304C
	IDX=0	MIC1305C
	DO 15 JJ=1,NACTIV	MIC1306C
	J=IACTIV(JJ)	MIC1307C
	IF((LS(J)-CLOCK-DUR(J)).LE.0) GO TO 15	MIC1308C
	IF(ANUM(J).LE.0) GO TO 15	MIC1309C
	IDX=IDX+1	MIC1310C
	IRESCH(IDX)=J	MIC1311C
15	CONTINUE	MIC1312C
C		MIC1313C
	IF(IRESCH(1).EQ.C) RETURN	MIC1314C
C		MIC1315C
C	CHECK IF ENOUGH RESOURCES CAN BE ACQUIRED	MIC1316C
	NN=1	MIC1317C
16	J=IRESCH(NN)	MIC1318C
C	NOT ENOUGH RES. FAIL	MIC1319C
		MIC1320C

IF(J.EQ.0) THEN	MIC13210
GO TO 10	MIC13220
ENDIF	MIC13230
MRES=ANUM(J)	MIC13240
DO 17 L=1,MRES	MIC13250
K=WCSHOP(ASHOP(J,L))	MIC13260
17 RNEED(K)=RNEED(K)-A(J,L)*Z(J)	MIC13270
MRES=ANUM(I)	MIC13280
ICHECK=0	MIC13290
DO 18 L=1,MRES	MIC13300
K=WCSHOP(ASHOP(I,L))	MIC13310
18 IF(RNEED(K).GT.0) ICHECK=1	MIC13320
C IF ICHECK=1, TRY MORE SLACK ACTIVITIES	MIC13330
IF(ICHECK.EQ.1) THEN	MIC13340
NN=NN+1	MIC13350
GO TO 16	MIC13360
ENDIF	MIC13370
C	MIC13380
C SUCCESS TO GET ENOUGH RESOURCES	MIC13390
C RELEASE RESOURCES OF NN SLACK ACT'S	MIC13400
DO 20 M=1,NN	MIC13410
J=IRESCH(M)	MIC13420
MRES=ANUM(J)	MIC13430
DO 20 L=1,MRES	MIC13440
K=WCSHOP(ASHOP(J,L))	MIC13450
AR(K)=AR(K)+A(J,L)*Z(J)	MIC13460
20 CONTINUE	MIC13470
C	MIC13480
C SCHEDULE ACTIVITY 1	MIC13490
ZNEW=ZU(I)	MIC13500
ZC=DUR(I)/(LS(I)-CLOCK)	MIC13510
IF(ZNEW.GT.ZC) ZNEW=ZC	MIC13520
MRES=ANUM(I)	MIC13530
DO 27 L=1,MRES	MIC13540
K=WCSHOP(ASHOP(I,L))	MIC13550
IF(A(I,L).LT..0001) GOTO 27	MIC13560
ZAR=AR(K)/A(I,L)	MIC13570
IF(ZNEW.GT.ZAR) ZNEW=ZAR	MIC13580
27 CONTINUE	MIC13590
C CHECK FLOW TRANSFER	MIC13600
IF(NP(I).EQ.0) GOTO 31	MIC13610
ISTART=PPOS(I)+1	MIC13620
IEND=NP(I)+PPOS(I)	MIC13630
DO 29 IPDS=ISTART,IEND	MIC13640
IF(CTRANP(IPDS).EQ.1) GOTO 29	MIC13650
IK=FRED(IPDS)	MIC13660
IF(TFINIS(IK).LE.CLOCK) GOTO 29	MIC13670
IF(DUR(I).LE.0) GOTO 29	MIC13680
ZMAXI=DUR(I)*(1.-CTRANP(IPDS))/(TFINIS(IK)-CLOCK)	MIC13690
IF(ZMAXI.LT.ZNEW) ZNEW=ZMAXI	MIC13700
29 CONTINUE	MIC13710
31 CONTINUE	MIC13720
C	MIC13730
IF(ZNEW.LT.ZL(I)) THEN	MIC13740
C CLEAR THE ADDED RESOURCES	MIC13750

	DC 32 M=1,NN	MIC13760
	J=IRESCH(M)	MIC13770
	MRES=ANUM(J)	MIC13780
	DC 32 L=1,MRES	MIC13790
	K=WCSHOP(ASHOP(J,L))	MIC13800
	AF(K)=AR(K)-A(J,L)*Z(J)	MIC13810
32	CONTINUE	MIC13820
	GO TO 10	MIC13830
	ENDIF	MIC13840
C		MIC13850
C	UPDATE AVAILABLE RESOURCES,ETC. AND SAVE I IN NSAVE	MIC13860
	DC 33 L=1,MRES	MIC13870
	K=WCSHOP(ASHOP(I,L))	MIC13880
	AR(K)=AR(K)-A(I,L)*ZNEW	MIC13890
33	CONTINUE	MIC13900
	ISAVE=ISAVE+1	MIC13910
	NSAVE(ISAVE)=I	MIC13920
	TSTART(I)=CLOCK	MIC13930
	TLU(I)=CLOCK	MIC13940
	CALL RESCHD(I,ZNEW,AR)	MIC13950
	WRITE(37,106) I	MIC13960
106	FORMAT(' ACTIVITY',I5,' IS SCHEDULED.')	MIC13970
C		MIC13980
C	UPDATE UNDOED ACTIVITIES AND SAVE J INTO MSAVE	MIC13990
	DC 35 M=1,NN	MIC14000
	J=IRESCH(M)	MIC14010
	WRITE(37,107) CLOCK,J,CLOCK+1.	MIC14020
107	FORMAT(1X,F6.2,' ACT.',I4,' IS DELAYED UNTIL',F6.2)	MIC14030
	Z(J)=0.	MIC14040
	PCV(J)=0.	MIC14050
	CALL CANCEL(J,IACTIV,NACTIV)	MIC14060
	MSAVE(JSAVE+M)=J	MIC14070
35	CONTINUE	MIC14080
	JSAVE=JSAVE+NN	MIC14090
C		MIC14100
10	CONTINUE	MIC14110
C		MIC14120
C	UPDATE ACTIVE AND READY LIST	MIC14130
	IF(ISAVE.EQ.0) RETURN	MIC14140
	DC 50 II=1,ISAVE	MIC14150
	I=NSAVE(II)	MIC14160
	CALL CANCEL(I,IREADY,NREADY)	MIC14170
	CALL ADDL(I,PRI,IACTIV,NACTIV)	MIC14180
50	CONTINUE	MIC14190
	IF(JSAVE.EQ.0) THEN	MIC14200
	WRITE(6,*)'**** LOGIC ERROR IN SUBROUTINE CRITIC.. CHECK ****'	MIC14210
	RETURN	MIC14220
	ENDIF	MIC14230
	DC 60 JJ=1,JSAVE	MIC14240
	J=MSAVE(JJ)	MIC14250
	CALL ADDL(J,PRI,IREADY,NREADY)	MIC14260
60	CONTINUE	MIC14270
C		MIC14280
	RETURN	MIC14290
	END	MIC14300

		MIC14310
		MIC14320
	SUBROUTINE RESCHD(NA,ZNEW,AR)	MIC14330
C		MIC14340
C	THIS ROUTINE RESETS TFINIS,Z,PCOV,TLU ACCORDING TO NEW INTENSITY	MIC14350
C		MIC14360
	DIMENSION DFL(30),PCV(2500),TLU(2500),TFINIS(2500),Z(2500)	MIC14370
	INTEGER DUR(2500)	MIC14380
	INTEGER FOLL(10000),NF(2500),FPOS(2500)	MIC14390
	INTEGER PRED(10000),NP(2500),PPOS(2500)	MIC14400
	DIMENSION TEVENT(10000)	MIC14410
	INTEGER TIMVEC(10000,2),ADTIME(1000,11)	MIC14420
	DIMENSION CTRANF(10000),CTRANP(10000),AK(200)	MIC14430
	COMMON/BB4/CTRANF,CTRANP	MIC14440
	COMMON/BB6/PRED,NP,PPOS	MIC14450
	COMMON/BB7/FOLL,NF,FPOS	MIC14460
	COMMON/BB8/PCV,TLU,TFINIS,Z,CLOCK,DUR,LENGTH	MIC14470
	COMMON/TTT/TEVENT,TIMVEC,ADTIME,NXTVEC	MIC14480
		MIC14490
	PCOV=1.	MIC14500
	ZOLD=Z(NA)	MIC14510
	IF(DUR(NA).LE.0) GOTO 1254	MIC14520
	PCOV=PCV(NA)+(CLOCK-TLU(NA))*Z(NA)/DUR(NA)	MIC14530
1254	K=1	MIC14540
	NPF=FPOS(NA)+1	MIC14550
	DFL(1)=1.	MIC14560
	IF(CTRANF(NPF).EQ.1.) GOTO 115	MIC14570
	K=2	MIC14580
	DFL(2)=CTRANF(NPF)	MIC14590
115	CONTINUE	MIC14600
C	ERASE OLD EVENT FROM SCHED LIST	MIC14610
	IF(TEVENT(5000+NA).EQ.0.) GOTO 201	MIC14620
	IF((DFL(2)-PCOV).LE.0.0001) GOTO 201	MIC14630
	CALL ERASE(5000+NA)	MIC14640
201	IF(TEVENT(2500+NA).EQ.0.) GOTO 202	MIC14650
	IF(PCOV.GE.0.9999) GO TO 202	MIC14660
	CALL ERASE(2500+NA)	MIC14670
C	RESCHEDULE NEW EVENTS	MIC14680
202	PCV(NA)=PCOV	MIC14690
	TLU(NA)=CLOCK	MIC14700
	IF(ZNEW.LE. .0001) THEN	MIC14710
	TFINIS(NA)=99999.	MIC14720
	WRITE(6,*)'ACT',NA,' INTERRUPTED AT',CLOCK	MIC14730
	GO TO 26	MIC14740
	ENDIF	MIC14750
	DO 120 J=1,K	MIC14760
	PLEFT=DFL(J)-PCOV	MIC14770
	IF(FLEFT.LE.0.0001) GOTO 120	MIC14780
	ANWTM=CLOCK+PLEFT*DUR(NA)/ZNEW	MIC14790
	IF(DFL(J).LT.1.)GOTO 122	MIC14800
	CALL SCHED(2500+NA,ANWTM)	MIC14810
121	TFINIS(NA)=ANWTM	MIC14820
	GOTO 120	MIC14830
122	CONTINUE	MIC14840
	CALL SCHED(5000+NA,ANWTM)	MIC14850

AD-A168 361

GUIDE TO SIMULATION SCHEDULING APPENDICES B AND C(U)
CALIFORNIA UNIV BERKELEY OPERATIONS RESEARCH CENTER
R C LEACHMAN ET AL. JAN 86 ORC-86-1-APP-B/C

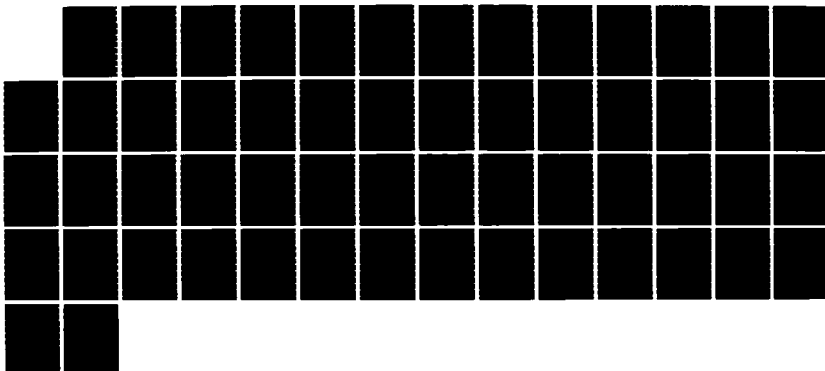
2/2

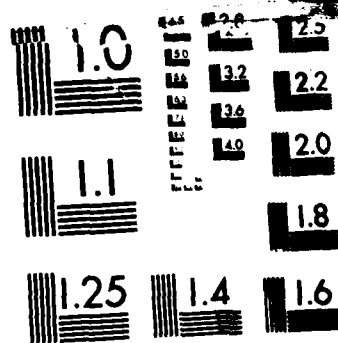
UNCLASSIFIED

N00014-76-C-0134

F/G 9/2

NL





MICROCOPY RESOLUTION TEST CHART
 NATIONAL BUREAU OF STANDARDS-1963-A

120	CONTINUE	MIC1486C
26	CONTINUE	MIC1487C
	Z(NA)=ZNEW	MIC1488C
C	WRITE(37,148) NA,Z(NA),PCOV,CLOCK,ZOLD	MIC1489C
C	WRITE(37,148) CLOCK,NA,Z(NA),ZOLD,PCCV,TFINIS(NA)	MIC1490C
C	+ ,AR(01),AR(02)	MIC1491C
	LENGTH=LENGTH+1	MIC1492C
C	148 FCMPAT(5X,15,5X,F6.3,5X,F6.3,5X,F8.2,5X,F6.3)	MIC1493C
	148 FCMPAT(1X,F6.2,' ',14,' ',F6.3,	MIC1494C
	+ ' ',F6.3,' ',F5.3,' ',F6.2,	MIC1495C
	+ ' ',F8.1)	MIC1496C
	RETURN	MIC1497C
	END	MIC1498C
	SUBROUTINE DWNGRD(M,ZLOW,Z,IIRUN,PR1)	MIC1499C
C		MIC1500C
C	THIS ROUTINE DOWNGRADES ACTIVITY INTENSITIES TO RECTIFY CAPACITY	MIC1501C
C	VIOLATIONS.	MIC1502C
	DIMENSION IREV(1000),NPRI(1000),A(2500,70),AR(200)	MIC1503C
	INTEGER ANUM(2500),ASHOP(2500,70),WCSHOP(200),NSHCP	MIC1504C
	INTEGER IACTIV(1000),IREADY(1000)	MIC1505C
	DIMENSION ZL(2500),Z(2500)	MIC1506C
	INTEGER PRI(2500)	MIC1507C
	COMMON/BB1/ IREADY, IACTIV,NREADY,NACTIV	MIC1508C
	COMMON/CC9/WCSHOP,NSHCP	MIC1509C
	COMMON/BB9/ANUM,ASHOP,A,AR	MIC1510C
	COMMON/BB10/IREV,NPRI,JFLAG	MIC1511C
	COMMON/BB11/ZL	MIC1512C
	DO 1162 I=1,M	MIC1513C
	J=NPRI(I)	MIC1514C
	JK=IREV(J)	MIC1515C
	IF(ZLOW.GT.ZL(JK)) ZLOW=ZL(JK)	MIC1516C
	ZMEX=Z(JK)-ZLOW	MIC1517C
C	DETERMINE IF ANY CONTRIBUTION DONE DUE TO JK	MIC1518C
	IFLAG=0	MIC1519C
	JFLAG=0	MIC1520C
	NRS=ANUM(JK)	MIC1521C
	IF(NRS.EQ.0) GOTO 1162	MIC1522C
	DO 1163 L=1,NRS	MIC1523C
	K=WCSHOP(ASHOP(JK,L))	MIC1524C
	AD=ZMEX*A(JK,L)	MIC1525C
	IF(AD.GT. .01 .AND. AR(K).LT.0.) IFLAG=1	MIC1526C
1163	CONTINUE	MIC1527C
	IF(IFLAG.EQ.0) GOTO 1162	MIC1528C
C	UPDATE AVAILABLE RESOURCES	MIC1529C
	DO 1246 L=1,NRS	MIC1530C
	K=WCSHOP(ASHOP(JK,L))	MIC1531C
1246	AR(K)=AR(K)+ZMEX*A(JK,L)	MIC1532C
	CALL RESCHD (JK,ZLOW,AR)	MIC1533C
C	CHECK IF CAPACITY VIOLATION RECTIFIED	MIC1534C
1233	CALL CHECK(AR,JFLAG)	MIC1535C
	IF(JFLAG.EQ.0) RETURN	MIC1536C
1162	CONTINUE	MIC1537C
C		MIC1538C
		MIC1539C
		MIC1540C

C	NEGATIVE AR =J LOGICAL ERROR	MIC1541C
	IF(IIRUN.EQ.2) THEN	MIC1542C
	WRITE(6,*)'NEGATIVE AR AFTER DISRUPTING .. LOGIC ERROR'	MIC1543C
	DC 1222 II=1,NSHOP	MIC1544C
	IF(AR(II).LT.0) THEN	MIC1545C
	WRITE(6,*)'I,AR=',II,AR(II)	MIC1546C
	ENDIF	MIC1547C
1222	CONTINUE	MIC1548C
	STOP	MIC1549C
	ENDIF	MIC1550C
	RETURN	MIC1551C
C	END	MIC1552C
		MIC1553C
		MIC1554C
		MIC1555C
	SUBROUTINE RAND(N,U)	MIC1556C
C		MIC1557C
C	THIS ROUTINE COMPUTES A RANDOM NUMBER 0< 'U' < 1 USING SEED 'N'	MIC1558C
C		MIC1559C
	N=N*843314661 + 453816693	MIC1560C
	IF(N.GE.0) GOTO 1	MIC1561C
	N=N + 2147483647 +1	MIC1562C
1	U=N* .4656612E-9	MIC1563C
	IF(U.LT.0. OR.U.GT.1) WRITE(6,*)' RAND OUT OF LIMIT',U	MIC1564C
	RETURN	MIC1565C
	END	MIC1566C
		MIC1567C
		MIC1568C
	SUBROUTINE SCHED(CODE,TIME)	MIC1569C
C		MIC1570C
C	THIS SUBROUTINE ADDS THE EVENT TO THE SCHEDULING LIST.	MIC1571C
C		MIC1572C
	INTEGER TIMVEC(10000,2),ADTIME(1000,11),NPCELL,NP2,CODE,LTIME	MIC1573C
	REAL TIME,TEVENT(10000)	MIC1574C
	COMMON/TTT/TEVENT,TIMVEC,ADTIME,NXTVEC	MIC1575C
	COMMON/YYY/LTIME	MIC1576C
		MIC1577C
	NPCELL=TIME*10+1+.5	MIC1578C
	IF(NPCELL.GT.LTIME) LTIME=NPCELL	MIC1579C
	IF(TIMVEC(NPCELL,1).NE.0) GOTO 1	MIC1580C
C	SCHEDULE FIRST EVENT AT 'TIME'	MIC1581C
	TIMVEC(NPCELL,1)=CODE	MIC1582C
	GOTO 7	MIC1583C
C		MIC1584C
1	IF(TIMVEC(NPCELL,2).NE.0) GOTO 2	MIC1585C
C	FIRST USE OF AUX. STORAGE AT 'TIME'	MIC1586C
	ADTIME(NXTVEC,1)=CODE	MIC1587C
	TIMVEC(NPCELL,2)=NXTVEC	MIC1588C
	GOTO 6	MIC1589C
2	NP2=TIMVEC(NPCELL,2)	MIC1590C
3	IF(ADTIME(NP2,10).NE.0) GOTO 5	MIC1591C
C	AUX. VECTOR DOESN'T POINT TO A NEXT ONE.	MIC1592C
	DC 4 I=1,9	MIC1593C
	IF(ADTIME(NP2,I).NE.0) GOTO 4	MIC1594C
	ADTIME(NP2,I)=CODE	MIC1595C

```

      GOTO 7
4  CONTINUE
C  EXTENSION VECTOR IS FULL. USE THE NEXT ONE AVAILABLE
    ADTIME(NP2,10)=NXTVEC
    ADTIME(NXTVEC,1)=CODE
C  FILL BACKWARD POINTER
    ADTIME(NXTVEC,11)=NP2
    GOTO 6
5  NP2=ADTIME(NP2,10)
    GOTO 3
6  NXT1=NXTVEC+1
C  FIND NEXT AVAILABLE AUX VECTOR
    DO 8 I=NXT1,1000
      IF(ADTIME(I,1).NE.0 ) GOTO 8
    NXTVEC=I
7  TEVENT(CODE)=TIME
    RETURN
8  CONTINUE
    WRITE (6,*) ' AUXILARRY EVENT STORAGE IS FULL. '
    STOP
    END

      SUBROUTINE RMV(CODE,TIME,LAST,CLOCK)
      -----
C  THIS SUBROUTINE FINDS THE THE NEXT EVENT IN THE LIST AND
C  ERASES IT. IF THERE ARE MORE THEN A SINGLE EVENT AT TIME,
C  RMV WILL PICK THE LAST ONE WHICH WAS SCHEDULED.
C  -----
      INTEGER FCELL,TIMVEC(10000,2),ADTIME(1000,11),CODE,LAST,LTIME
      REAL TIME,CLOCK,STOPTM,TEVENT(10000)
      COMMON/RMVMN/TEND
      COMMON/TIT/TEVENT,TIMVEC,ADTIME,NXTVEC
      COMMON/YYY/LTIME

      LAST=0
      FCELL=CLOCK*10+1+.5
      DO 10 I=FCELL,LTIME
        IF(TIMVEC(I,1).EQ.0) GOTO 10
        IF(TIMVEC(I,2).NE.0) GOTO 1
C  SINGLE EVENT AT 'TIME'
        CODE=TIMVEC(I,1)
C  TIME=FLCAT(I-1)/10.      MODIFIED TO USE EXACT TIME VALUE
        TIME=TEVENT(CODE)
        TIMVEC(I,1)=0
        LAST=1
        GOTO 32
C
C  MORE THEN ONE EVENT AT 'TIME'
1  NP2=TIMVEC(I,2)
2  IF(ADTIME(NP2,10).NE.0) GO TO 5
    DO 4 J=2,10
      IF(ADTIME(NP2,J).NE.0) GOTO 4
      CODE = ADTIME(NP2,J-1)

```

MIC1596
 MIC1597
 MIC1598
 MIC1599
 MIC1600
 MIC1601
 MIC1602
 MIC1603
 MIC1604
 MIC1605
 MIC1606
 MIC1607
 MIC1608
 MIC1609
 MIC1610
 MIC1611
 MIC1612
 MIC1613
 MIC1614
 MIC1615
 MIC1616
 MIC1617
 MIC1618
 MIC1619
 MIC1620
 MIC1621
 MIC1622
 MIC1623
 MIC1624
 MIC1625
 MIC1626
 MIC1627
 MIC1628
 MIC1629
 MIC1630
 MIC1631
 MIC1632
 MIC1633
 MIC1634
 MIC1635
 MIC1636
 MIC1637
 MIC1638
 MIC1639
 MIC1640
 MIC1641
 MIC1642
 MIC1643
 MIC1644
 MIC1645
 MIC1646
 MIC1647
 MIC1648
 MIC1649
 MIC1650

```

C      TIME=(I-1)/10.                                MIC1651C
C      TIME=TEVENT(CODE)                              MIC1652C
C      ADTIME(NP2,J-1)=0                              MIC1653C
C      ERASE THE THE LAST EVENT AT 'TIME'. IF IT IS THE ONLY ONE MIC1654C
C      IN THE AUXILLARY VECTOR, RELEASE THE VECTOR.    MIC1655C
C      IF(J.NE.2) GOTO 32                              MIC1656C
C      NP22=ADTIME(NP2,11)                             MIC1657C
C      IF(NP22.NE.0) GOTO 3                             MIC1658C
C      TIMVEC(I,2)=0                                    MIC1659C
C      GOTO 31                                          MIC1660C
3     ADTIME(NP2,11)=0                                  MIC1661C
C      ADTIME(NP22,10)=0                               MIC1662C
31    IF(NXTVEC.GT.NP2) NXTVEC=NP2                     MIC1663C
32    IF(TIME.GE.TEND) GOTO 11                          MIC1664C
C      RETURN                                          MIC1665C
4     CONTINUE                                          MIC1666C
5     NP2=ADTIME(NP2,10)                               MIC1667C
C      GOTO 2                                          MIC1668C
10    CONTINUE                                          MIC1669C
C      WRITE(6,*) 'SCHEDULER TABLE IS EMPTY'        MIC1670C
C      LAST=2                                          MIC1671C
C      RETURN                                          MIC1672C
11    WRITE(6,*) 'TIME LIMIT EXCEEDED. TIME IS ',TIME,' DAYS' MIC1673C
C      WRITE(6,*) 'CODE=',CODE                      MIC1674C
C      LAST=3                                          MIC1675C
C      END                                            MIC1676C
C
C      SUBROUTINE ERASE(CODE)                          MIC1677C
C      -----                                         MIC1678C
C      THIS SUBROUTINE IS ERASING EVENTS FROM THE SCHEDULER LIST MIC1679C
C      -----                                         MIC1680C
C
C      INTEGER TIMVEC(10000,2),ADTIME(1000,11),CODE,CODE1 MIC1681C
C      REAL TIME,TEVENT(10000)                       MIC1682C
C      COMMON/ITT/TEVENT,TIMVEC,ADTIME,NXTVEC          MIC1683C
C
C      TIME=TEVENT(CODE)                               MIC1684C
C      IF(TIME.EQ.0.) RETURN                           MIC1685C
C
C      *** EVENT HADN'T BEEN SCHEDULED                MIC1686C
C      TEVENT(CODE)=0.                                 MIC1687C
C      NCELL=TIME*10+1 +.5                             MIC1688C
C
C      THE TRANSFORMATION IS AS FOLLOWS                MIC1689C
C      TIME*10 +1 ( FOR T=0. ) +.5 ( FOR ROUNDING.) MIC1690C
C      IF(TIMVEC(NCELL,1).NE.CODE) GOTO 1              MIC1691C
C      CALL RMV(CODE1,TIME,L,TIME)                     MIC1692C
C      IF(CODE.EQ.CODE1) RETURN                         MIC1693C
C      TIMVEC(NCELL,1)=CODE1                           MIC1694C
C      RETURN                                          MIC1695C
1     NP2=TIMVEC(NCELL,2)                             MIC1696C
2     DO 3 J=1,9                                       MIC1697C
C      IF(ADTIME(NP2,J).NE.CODE) GOTO 3                MIC1698C
C      CALL RMV(CODE1,TIME,L,TIME)                     MIC1699C
C      IF(CODE1.EQ.CODE) RETURN                         MIC1700C
C      ADTIME(NP2,J)=CODE1                             MIC1701C
3     CONTINUE                                          MIC1702C

```

```

NP2=ADTIME(NP2,10)
IF(NP2.GT.0) GOTO 2
RETURN
END

```

```

SUBROUTINE ACCUM(TIAS,WCSHOP,NSHOP)

```

```

-----
THIS SUBROUTINE ACCUMULATES THE USAGE OF THE RESOURCES.
-----

```

```

INTEGER WCSHOP(200),NSHOP,ANUM(2500),ASHOP(2500,70),IACTIV(1000)
INTEGER IREADY(1000),DUR(2500)
REAL ACUMRS(200),ACUMSP(50),A(2500,70),CLOCK,TIAS,Z(2500)
REAL PCV(2500),TLU(2500),TFINIS(2500),AF(200)
COMMON/BB1/IREADY,IACTIV,NREADY,NACTIV
COMMON/BB8/PCV,TLU,TFINIS,Z,CLOCK,DUR,LENGTH
COMMON/BB9/ANUM,ASHOP,A,AR
COMMON/CC1C/ACUMRS,ACUMSP

```

```

C IF(TIAS.EQ.CLOCK) RETURN
DO 1 I=1,NACTIV
  II=IACTIV(I)
  NRS=ANUM(II)
  IF(NRS.EQ.0) GOTO 1
  DO 2 J=1,NRS
    DRES=A(II,J)*(CLOCK-TIAS)*Z(II)
    L=ASHOP(II,J)
    ACUMRS(L)=ACUMRS(L)+DRES
    ACUMSP(WCSHOP(L))=ACUMSP(WCSHOP(L))+DRES
  2 CONTINUE
1 CONTINUE
TIAS=CLOCK
RETURN
END

```

```

SUBROUTINE DCCNV(WKD,CLND)

```

```

C *****
C *
C * THIS ROUTINE CONVERTS THE WORKING DAY INTO CALENDAR DATE
C *
C *****

```

```

INTEGER WKD,YEAR,SMON,SDAY,SYR,WKDTB(120,31),CLND(3)
COMMON/DCCN/SMON,SDAY,SYR,YEAR,WKDTB

```

```

C IF(WKD.LT.0) THEN
  WRITE(6,*) '* ILLEGAL WORKING DATE =',WKD
  CLND(1)=0
  CLND(2)=0
  CLND(3)=0
  RETURN
ENDIF

```

```

C IF(WKD.GE.99998) THEN
  CLND(1)=99

```

```

MIC1706C
MIC1707C
MIC1708C
MIC1709C
MIC1710C
MIC1711C
MIC1712C
MIC1713C
MIC1714C
MIC1715C
MIC1716C
MIC1717C
MIC1718C
MIC1719C
MIC1720C
MIC1721C
MIC1722C
MIC1723C
MIC1724C
MIC1725C
MIC1726C
MIC1727C
MIC1728C
MIC1729C
MIC1730C
MIC1731C
MIC1732C
MIC1733C
MIC1734C
MIC1735C
MIC1736C
MIC1 370
MIC1 380
MIC1739C
MIC1740C
MIC1741C
MIC1742C
MIC1743C
MIC1744C
MIC1745C
MIC1746C
MIC1747C
MIC1748C
MIC1749C
MIC1750C
MIC1751C
MIC1752C
MIC1753C
MIC1754C
MIC1755C
MIC1756C
MIC1757C
MIC1758C
MIC1759C
MIC1760C

```

CLND(2)=99	MIC1761C
CLND(3)=99	MIC1762C
RETURN	MIC1763C
ENDIF	MIC1764C
C	MIC1765C
ISYR=(SYR-YEAR)*12+SMON	MIC1766C
M=WKD+1	MIC1767C
DO 310 JI=SDAY,31	MIC1768C
M=M-WKDTB(ISYR,JI)	MIC1769C
JCUT=JI	MIC1770C
IF(M.EQ.0)GOTO 380	MIC1771C
310 CONTINUE	MIC1772C
JSYR=ISYR+1	MIC1773C
320 DO 330 JJ=1,31	MIC1774C
M=M-WKDTB(JSYR,JJ)	MIC1775C
JCUT=JJ	MIC1776C
IF(M.EQ.0)GOTO 390	MIC1777C
330 CONTINUE	MIC1778C
JSYR=JSYR+1	MIC1779C
GOTO 320	MIC1780C
380 CLND(1)=SMON	MIC1781C
CLND(2)=JCUT	MIC1782C
CLND(3)=SYR	MIC1783C
GOTO 300	MIC1784C
390 KMON=MOD(JSYR,12)	MIC1785C
IF(KMON.GT.0)GOTO 395	MIC1786C
KMON=12	MIC1787C
395 KYR=(JSYR-KMON)/12+YEAR	MIC1788C
CLND(1)=KMON	MIC1789C
CLND(2)=JCUT	MIC1790C
CLND(3)=KYR	MIC1791C
300 CONTINUE	MIC1792C
C	MIC1793C
CLND(3)=CLND(3)-1900	MIC1794C
C	MIC1795C
RETURN	MIC1796C
END	MIC1797C
	MIC1798C
	MIC1799C
	MIC1800C
SUBROUTINE CALA(CLND,WD)	MIC1801C
C *****	MIC1802C
C *	MIC1803C
C * THIS ROUTINE CONVERTS THE CALENDAR DATE INTO WORKING DAYS	MIC1804C
C *	MIC1805C
C *****	MIC1806C
INTEGER WD, YEAR, SMON, SDAY, SYR, WKDTB(120,31), CLND(3)	MIC1807C
COMMON/DCON/SMON, SDAY, SYR, YEAR, WKDTB	MIC1808C
C	MIC1809C
CLND(2)=CLND(2)-1	MIC1810C
C	MIC1811C
ISYR=(SYR-YEAR)*12+SMON	MIC1812C
WD=0	MIC1813C
IYR=(CLND(3)-YEAR)*12+CLND(1)	MIC1814C
IF(IYR.LT.ISYR)GOTO 650	MIC1815C

IF(IYR.EQ.ISYR)GOTO 590	MIC18160
DC 400 PI=ISYR,IYR	MIC18170
IF(MI.EQ.ISYR)GOTO 410	MIC18180
IF(MI.EQ.IYR)GOTO 420	MIC18190
DC 440 PK=1,31	MIC18200
WD=WD+WKDTB(MI,MK)	MIC18210
440 CONTINUE	MIC18220
GOTO 400	MIC18230
420 LM=CLND(2)	MIC18240
DC 430 ML=1,LM	MIC18250
WD=WD+WKDTB(MI,ML)	MIC18260
430 CONTINUE	MIC18270
GOTO 400	MIC18280
410 DC 450 MJ=SDAY,31	MIC18290
WD=WD+WKDTB(MI,MJ)	MIC18300
450 CONTINUE	MIC18310
400 CONTINUE	MIC18320
GOTO 470	MIC18330
580 LK=CLND(2)	MIC18340
IF(LK.LI,SDAY)GOTO 650	MIC18350
DC 460 LI=SDAY,LK	MIC18360
WD=WD+WKDTB(IYR,LI)	MIC18370
460 CONTINUE	MIC18380
470 REP=WD	MIC18390
GOTO 500	MIC18400
C	MIC18410
650 PRINT *, ' ILLEGAL TARGET FINISH ** PLEASE CHECK '	MIC18420
REP = 0	MIC18430
500 CONTINUE	MIC18440
C	MIC18450
RETURN	MIC18460
END	MIC18470

```

C *****
C
C          BMICORG  FORTRAN
C          -----
C    BMICORG DOES SCHEDULING ACCORDING TO 3 CATEGORIES :
C    1. DETERMINISTIC SCHEDULING
C    2. STOCHASTIC SCHEDULING REGARDING TEST REWORK LOOPS
C    3. SIMULATION USING RANDOM RESOURCE REQUIREMENTS
C    ALL 3 CAT. CAN USE UPGRADING OR UP & DOWNGRADING ALGORITHM
C    * VERSION FOR BATCH RUN
C *****

CHARACTER*50 NAME
INTEGER TOTMHR(50), TMILE(50), ACSHP(50), ACCMC(200),
+ TOTMDY(300), IMILE(3), SMILE(50), MDATE(3)
REAL RESREG(50), ATOT(300)
DIMENSION A(2500,70), ITYPE(2500)
INTEGER ASHP(2500,70), ANUM(2500), MILE(2500), MILSTN(50)
INTEGER FROM(2500), TO(2500)
DIMENSION ZL(2500), ZU(2500), Z(2500)
DIMENSION CAP(200), AR(200), LOST(15), DSEEDS(5), A1(2500,70)
DIMENSION PCV(2500), TLU(2500)
DIMENSION TSTART(2500), TFINIS(2500), ATFIN(2500)
DIMENSION CIC(50,200)
REAL LS(2500), DFL(30), ACUMRS(200), ACLMSP(50)
INTEGER FOLL(10000), NF(2500), FPOS(2500), WCSHCP(200), NSHCP
INTEGER PRED(10000), NP(2500), PPOS(2500), REPDAT(50), NREPUT
REAL CTRANF(10000), CTRANP(10000)
INTEGER PRI(2500), DUR(2500), RANK(2500), TFCC(50)
INTEGER STATUS(2500), IACTIV(1000), IREADY(1000)
DIMENSION NPRI(1000), DRV(1000), IREV(1000), TEVENT(10000)
INTEGER CODE, TIMVEC(10000,2), ADTIME(1000,11)
INTEGER TFCLL(250,2), TRANK(2500), DURR(600,5)
DIMENSION TATTR(250,5), PROB(10), RDUR(10), EDUR(2500), PROBR(600,5)
INTEGER YEAR, SMON, SDAY, SYR, WKDTB(120,31), CALND1(3), CALND2(3)

COMMON/BB1/ IREADY, IACTIV, NREADY, NACTIV
COMMON/BB3/ JNX, JF, JL, LWC
COMMON/BB4/ CTRANF, CTRANP
COMMON/BB5/ LS, NNODE, RANK, ITURN
COMMON/BB6/ PRED, NP, PPOS
COMMON/BB7/ FOLL, NF, FPOS
COMMON/BB8/ PCV, TLU, TFINIS, Z, CLOCK, DUR, LENGTH
COMMON/BB9/ ANUM, ASHP, A, AR
COMMON/BB10/ IREV, NPRI, JFLAG
COMMON/BB11/ ZL
COMMON/BB12/ ITYPE, ZU, CAP, TFCC, CIC
COMMON/BB13/ AKES, NCC
COMMON/CC9/ WCSHCP, NSHCP
COMMON/CC10/ ACUMRS, ACUMSP
COMMON/RMVMH/ TEND
COMMON/YYY/ LTIME
COMMON/TTT/ TEVENT, TIMVEC, ADTIME, NXTVEC

```

BP10001
 BP10002
 BP10003
 BP10004
 BP10005
 BP10006
 BP10007
 BP10008
 BP10009
 BP10010
 BP10011
 BP10012
 BP10013
 BP10014
 BP10015
 BP10016
 BP10017
 BP10018
 BP10019
 BP10020
 BP10021
 BP10022
 BP10023
 BP10024
 BP10025
 BP10026
 BP10027
 BP10028
 BP10029
 BP10030
 BP10031
 BP10032
 BP10033
 BP10034
 BP10035
 BP10036
 BP10037
 BP10038
 BP10039
 BP10040
 BP10041
 BP10042
 BP10043
 BP10044
 BP10045
 BP10046
 BP10047
 BP10048
 BP10049
 BP10050
 BP10051
 BP10052
 BP10053
 BP10054
 BP10055

FILE: BPICORG FCRTRAN A VM/SP CMS RELEASE 3.1.E 851112, CFC - U.C. BERKELEY

```
COMMON/DCUN/SMON,SDAY,SYR,YEAR,WKDTB
CHARACTER*1 NNTES
C READ INPUT FILES
C READ(1,323) NAME,
+ IRAPOL,ILOW,IUP,NTEST,KTEND,NCYCLE,ITARG,ICALD
C READ(33,322) NRES,NSHOP
READ(34,322) NCC
READ(35,322) NREPDT
322 FCRPAT(20X,I5)
C READ SEED FOR RANDOM NUMBER
READ(17,1777,END=777) NRAND
GC TO 778
777 WRITE(6,*) ' *** RANDOM SEED FILE NOT FOUND. CHECK BBRAND DATA'
STOP
778 CONTINUE
C WRITE(6,*) ' *****'
WRITE(6,*) ' * M I C R O R G (BATCH) *'
WRITE(6,*) ' *****'
WRITE(6,*) ' '
IF(NCYCLE.EQ.1) ISIM=1
IF(NCYCLE.NE.1) ISIM=2
C WRITE(6,*) ' THANK YOU. IN PROCESSING.....'
C ZLOW=ILCH*.01
ZUP=IUP*.01
TIME=0.
TEND=KTEND*.1.
C WRITE(6,*) ' READING FOLL & PRED'
READ(2,951) FOLL
READ(3,951) PRED
WRITE(6,*) ' READING ACTIVITY-RESOURCE DATA'
READ(4,951) ANUM
WRITE(6,*) ' READING TRANSFER COEFFICIENTS'
C INITIALIZE COEFFICIENTS TO 1.
DC 344 I=1,10000
CTRAF(I)=1.
CTRANP(I)=1.
344 CONTINUE
READ(12,952,END=7) CTRAF,CTRANP
7 WRITE(6,*) ' READING ACTIVITIES DATA'
NMILE=0
ITURN=0
DC 950 NA=1,2500
READ(7,952,END=101) FROM(NA),TO(NA),OLR(NA),ITYPE(NA)
IF(ITYPE(NA).NE.3.AND.ITYPE(NA).NE.4) ITURN=ITURN+1
IF(ITYPE(NA).NE.7) GOTO 954
NMILE=NMILE+1
MILE(NA)=NMILE
BP100560
BP100570
BP100580
BP100590
BP100600
BP100610
BP100620
BP100630
BP100640
BP100650
BP100660
BP100670
BP100680
BP100690
BP100700
BP100710
BP100720
BP100730
BP100740
BP100750
BP100760
BP100770
BP100780
BP100790
BP100800
BP100810
BP100820
BP100830
BP100840
BP100850
BP100860
BP100870
BP100880
BP100890
BP100900
BP100910
BP100920
BP100930
BP100940
BP100950
BP100960
BP100970
BP100980
BP100990
BP101000
BP101010
BP101020
BP101030
BP101040
BP101050
BP101060
BP101070
BP101080
BP101090
BP101100
```

MILSTN(NMILE)=FROM(NA)	BP10111
954 READ(8,961) FPOS(NA), NF(NA), PPOS(NA), NP(NA)	BP10112
K=ANUM(NA)	BP10113
IF(K.EQ.0) GO TO 950	BP10114
READ(9,951) (ASHOP(NA,J),J=1,K)	BP10115
READ(10,963) (AI(NA,J),J=1,K)	BP10116
950 CCNTINUE	BP10117
101 NNODE=NA-1	BP10118
C	BP10119
C READ MAPPING WC >> SHOP.	BP10120
C READ(13,951) (WCSHOP(I),I=1,NRES)	BP10121
C READ DATES OF REPORTING DATES	BP10122
IF(NREPD.T.GT.0)	BP10123
* READ(16,951) (REPDAT(I),I=1,NREPD.T)	BP10124
C	BP10125
C IF THERE IS NO TEST ACT. SKIP READING	BP10126
IF(NTST.EQ.0) GOTU 379	BP10127
WRITE(6,*) ' READING REWORK-ACTIVITIES DATA '	BP10128
READ(18,951) TFOLL	BP10129
READ(19,953) TATT	BP10130
READ(30,951) DURR	BP10131
READ(31,953) PROBR	BP10132
READ(20,951) TRANK	BP10133
379 CONTINUE	BP10134
WRITE(6,*) ' READING CAPACITY(TIME,RESOURCE) '	BP10135
DC 497 I=1,NCC	BP10136
READ(14,3001,END =497) (CIC(I,J),J=1,ASHOP)	BP10137
497 CONTINUE	BP10138
IF(NCC.LE.1) GOTO 387	BP10139
C READ TIME OF CAPACITY CHANGES.	BP10140
READ (15,951,END=387) (TFCC(I),I=1,NCC)	BP10141
387 CONTINUE	BP10142
WRITE(6,*) ' READING RANK OF ACTIVITIES '	BP10143
READ(11,951) (RANK(I),I=1,NNODE)	BP10144
C	BP10145
C PREPARE CONVERTING WORKING DATES INTO CALENDAR DATES	BP10146
C AND VICE VERSA	BP10147
C READ THE STARTING DATE OF THE PROJECT	BP10148
READ(52,520)SMON,SDAY,SYR	BP10149
520 FORMAT(10X,3I5)	BP10150
C READ THE YEAR OF BEGINNING IN THE WORKING TABLE	BP10151
READ(53,530)YEAR	BP10152
530 FORMAT(10X,I5)	BP10153
C READ THE WORKING DATES TABLE FROM FILE WORKING DATES	BP10154
DC 533 I=1,120	BP10155
READ(54,540)(WKDTB(I,J),J=1,31)	BP10156
533 CONTINUE	BP10157
540 FORMAT(3I12)	BP10158
C	BP10159
C TRANSLATE CALENDAR DATE OF TARGET FINISH INTO WORKING DAY	BP10160
C	BP10161
IF(ITARG.EQ.0) GO TO 551	BP10162
IYER=ITARG/100	BP10163
IMON=ITARG/10000	BP10164
MCATE(3)=ITARG-IYER*100+1900	BP10165

```

MDATE(1)=IMON
MDATE(2)=(ITARG-IMON*10000-MDATE(3)+1900)/100
CALL CALA(MDATE,ISCFIN)
551 CONTINUE
C
C
C   END OF DATA
C   -----
C   WRITE(6,*) ' END OF DATA . INITIALIZE VALUES FOR SIMULATION '
C   -----
C   FIND EXPECTED DURATIONS FOR TEST ACTIVITIES
C
C   IF ( ITURN .EQ. NNODE ) GO TO 2999
C   REPAIR ACTIVITIES
K=ITURN +1
DO 84 I=K,NNODE
  M=I-ITURN
  DO 83 N=1,5
    PROB(N)=PROBR(M,N)
83    RDUR(N)=CURR(M,N)
84 DLR(I)=XMEAN(PROB,RDUR,5)
2999 NFLAG=0
DO 383 I=1,NNODE
  IF(ITYPE(I).NE.2) GOTO 381
C   FIND EXPECTED DURATIONS OF REPAIR AFTER TEST
J=TRANK(I)
IF(TFOLL(J,1).EQ.0.AND.TFOLL(J,2).EQ.0) ITYPE(I)=C
IF(ITYPE(I).NE.2) GOTO 381
DO 384 L=1,2
  NC=0
  PROB(L)=TATTR(J,3+L)
  K=TFOLL(J,L)
  RDUR(L)=C.
  IF(K.EQ.C) GOTO 384
  RDUR(L)=CUR(K)
385  IS=FPCS(K)+1
  M=FOLL(IS)
  NC=NC+1
  IF(TO(I).EQ.TO(K)) GOTO 384
  K=FOLL(IS)
  PCUR(L)=KRDUR(L)+DUR(K)
  IF(NC.LE.4) GOTO 385
C
  WRITE (6,94) I,J
94  FORMAT(5X,'TEST',3X,14,' RANKED',15,3X,' IS UNCONNECTED')
  NFLAG=1
  GOTO 383
384 CONTINUE
PR=TATTR(J,1)
ALFA=TATTR(J,2)
EDUR(I)=DUR(I)+(ENS(PR,ALFA)-1.)*DUR(I)+XMEAN(PROB,RDUR,5)
C   EDUR(I)=DUR(I)+(1.-PR)*XMEAN(PROB,RDUR,2)
GOTO 393
381 EDUR(I)=DUR(I)

```

```

BP10166
BP10167
BP10168
BP10169
BP10170
BP10171
BP10172
BP10173
BP10174
BP10175
BP10176
BP10177
BP10178
BP10179
BP10180
BP10181
BP10182
BP10183
BP10184
BP10185
BP10186
BP10187
BP10188
BP10189
BP10190
BP10191
BP10192
BP10193
BP10194
BP10195
BP10196
BP10197
BP10198
BP10199
BP10200
BP10201
BP10202
BP10203
BP10204
BP10205
BP10206
BP10207
BP10208
BP10209
BP10210
BP10211
BP10212
BP10213
BP10214
BP10215
BP10216
BP10217
BP10218
BP10219
BP10220

```

383	CONTINUE	BP10221
	IF(NFLAG.EQ.1) STOP	BP10222
C	SET LOWER AND UPPER BOUND OF INTENSITY	BP10223
	DC 970 I=1,NNODE	BP10224
	ZL(I)=ZLOW	BP10225
	ZU(I)=ZUP	BP10226
	IF(ITYPE(I).EQ.1.OR.ITYPE(I).EQ.2.OR.ITYPE(I).EQ.4) THEN	BP10227
	ZL(I)=1.0	BP10228
	ZU(I)=1.0	BP10229
	ENDIF	BP10230
970	CONTINUE	BP10231
C		BP10232
C	START SIMULATION CYCLIES	BP10233
C	-----	BP10234
	DO 1999 ICYCLE=1,NCYCLE	BP10235
C		BP10236
	NXTVEC=1	BP10237
	TIME=0.	BP10238
	LTIME=0	BP10239
	CLOCK=0.	BP10240
	TACCUM=0.	BP10241
	TIAS=0.	BP10242
	LAST=0	BP10243
	ISWICH=0	BP10244
	NREC=0	BP10245
	PFINIS=0.	BP10246
	NACTIV=0	BP10247
	NREADY=0	BP10248
	LENGTH=0	BP10249
		BP10250
C	RESET SCHEDULE TABLES	BP10251
	DO 432 I=1,10000	BP10252
	TIMVEC(I,1)=0	BP10253
	TIMVEC(I,2)=0	BP10254
432	TEVENT(I)=0.	BP10255
	DO 434 I=1,1000	BP10256
	DO 434 J=1,11	BP10257
434	ACTIME(I,J)=0	BP10258
	DO 1778 L=1,NMILE	BP10259
1778	TMILE(L)=99999	BP10260
		BP10261
C	RESET ACCUMULATION	BP10262
	DC 920 I=1,NRES	BP10263
920	ACUMRS(I)=0.	BP10264
	DC 921 I=1,NSHUP	BP10265
	TCTPHR(I)=0.	BP10266
921	ACUMSP(I)=0.	BP10267
		BP10268
C	FIND THE ACTUAL DURATION FOR REPAIR ACTIVITIES.	BP10269
	K=ITURN+1	BP10270
	DO E6 I=K,NNODE	BP10271
	M=1-ITURN	BP10272
	DC E9 N=1,5	BP10273
E9	PRUE(N)=PR(PR(M,N)	BP10274
	CALL PICK(PFCE,KM,5)	BP10275

86	DUR(I)=DURR(M,KM)	BP102760
C	RANDOMIZE THE INPUT REQUIREMENTS	BP102770
	DO 1740 I=1,NNODE	BP102780
	NRS=ANUP(I)	BP102790
	IF(NRS.EQ.0) GOTO 1740	BP102800
	IF(ITYPE(I).GT.4) GOTO 1740	BP102810
	DO 1740 J=1,NRS	BP102820
	IF(ISIM.EQ.2) THEN	BP102830
	CALL RAND(NRAND,URAND)	BP102840
	A(I,J)=AI(I,J)*(URAND*.4 + .8)	BP102850
	ENDIF	BP102860
	IF(ISIM.EQ.1) A(I,J)=AI(I,J)	BP102870
1740	CONTINUE	BP102880
C	SET RESURCE CAPACITIES	BP102890
C		BP102900
	DO 483 L=1,NSHOP	BP102910
483	CAP(L)=CIC(1,L)	BP102920
C		BP102930
C	INITIALIZE ACTIVITY DATA	BP102940
	DO 382 I=1,NNODE	BP102950
	TFINIS(I)=99999.	BP102960
	ATFIN(I)=99999.	BP102970
	STATUS(I)=NP(I)	BP102980
	Z(I)=0.	BP102990
	PCV(I)=0.	BP103000
	TLC(I)=0.	BP103010
382	CONTINUE	BP103020
C		BP103030
C	START SIMULATION	BP103040
C		BP103050
	WRITE(6,*) '## START SIMULATION'	BP103060
	WRITE(6,*) '## RUN CYCLE = ',ICYCLE	BP103070
C		BP103080
C	CALCULATE ES AND LS BY CPM AND SET INITIAL PRIORITIES	BP103090
C		BP103100
	CALL LSCHED(EDUR,TFINIS,ISCFIN,ITYPE)	BP103110
	CALL SORT(LS,PRI,NNODE,'NWRITE')	BP103120
C		BP103130
C	SCHEDULE CAPACITY CHANGES	BP103140
	IF(NCC.LE.1) GOTO 313	BP103150
	DO 543 I=2,NCC	BP103160
	TM=TFCC(I)	BP103170
543	CALL SCHED(7500 + I, TM)	BP103180
C		BP103190
C	SCHEDULE REPORTING DATES	BP103200
313	IF(NREPDT.EQ.0) GOTO 314	BP103210
	DO 544 I=1,NREPDT	BP103220
	TM=FEPDAT(I)	BP103230
	CALL SCHED(8000 + I, TM)	BP103240
544	CONTINUE	BP103250
314	CONTINUE	BP103260
C	INITIALIZE AVAILABLE RESOURCES	BP103270
	DO 132 J=1,NSHOP	BP103280
132	AR(J)=CAP(J)	BP103290
		BP103300

C	INITIALIZE INTENSITY MATRIX	BM10331
	DC 244 I=1,NNODE	BM10332
	Z(I)=0.	BM10333
	PCV(I)=0.	BM10334
244	TLU(I)=0.	BM10335
C		BM10336
C	PUT ACTIVITIES WITHOUT PREDECESSOR INTO READY LIST	BM10337
	NACTIV=0	BM10338
	NREADY=0	BM10339
	DO 112 I=1,ITURN	BM10340
	IF(NP(I).NE.0) GO TO 112	BM10341
C	IF(ITYPE(I).GT.4) GO TO 2112	BM10342
	IF(ITYPE(I).GE.3) GO TO 2112	BM10343
	CALL ADDL(I,PRI,IREADY,NREADY)	BM10344
	GO TO 112	BM10345
2112	TSTART(I)=0.	BM10346
	TFINIS(I)=0.	BM10347
C	IF(ITYPE(I).EQ.5) TFINIS(I)=DUR(I)	BM10348
	CALL ADDL(I,PRI,IACTIV,NACTIV)	BM10349
	CALL SCHED(2500+I,TFINIS(I))	BM10350
	Z(I)=ZU(I)	BM10351
112	CONTINUE	BM10352
C		BM10353
C	***** START SIMULATION *****	BM10354
C		BM10355
	GOTO 810	BM10356
C	GET THE NEXT EVENT FROM THE LIST	BM10357
C		BM10358
41	CALL RMV(CCODE,TIME,LAST,CLOCK)	BM10359
	ELTP=TIME-CLOCK	BM10360
	CLOCK=TIME	BM10361
	IF(LAST.EQ.3) WRITE(6,*)'TIME LIMIT EXCEEDED'	BM10362
	+ , 'TIME=',CLOCK, ' CODE=',CODE	BM10363
	IF(LAST.GE.2) GOTO 91	BM10364
C		BM10365
C	CCODE= C+ : START DELAYED FOLLOWER OF A MILESTONE.	BM10366
C	CCODE=2500+ : AN ACTIVITY HAS FINISHED. START FOLLOWERS.	BM10367
C	CCODE=5000+ : FLOW-TRANSFERRED FOLLOWER START	BM10368
C	CCODE=7500+ : CAPACITY IS CHANGED.	BM10369
C	CCODE=8000+ : REPORTING DATE	BM10370
C		BM10371
	IF(CODE.LT.2500) GOTO 51	BM10372
	IF(CODE.LT.5000) GOTO 61	BM10373
	IF(CODE.LT.7500) GOTO 71	BM10374
	IF(CODE.LT.8000) GOTO 222	BM10375
C		BM10376
C	REPORTING DATE ... REPORT RESOURCE AVERAGES	BM10377
C		BM10378
	WRITE(6,*) 'REPORTING EVENT. TIME IS',CLOCK	BM10379
	CALL ACCUM(TIAS,WCSHOP,NSHOP)	BM10380
710	TIAS=CLOCK	BM10381
	DELT=CLOCK-TACCU	BM10382
	IF(DELT.EQ.0.) GOTO 713	BM10383
C		BM10384
C	CALCULATE AVG. RESOURCES USED BETWEEN DELT	BM10385

DO 711 IRES=1,NRES	BM10386
711 ACCWC(IRES)=ACUMRS(IRES)/DELT	BM10387
DO 712 ISHOP=1,NSHOP	BM10388
TOTMHR(ISHOP)=TCTMHR(ISHOP)+ACUMSP(ISHOP)	BM10389
712 ACSHOP(ISHOP)=ACUMSP(ISHOP)/DELT	BM10390
TACCUM=CLOCK	BM10391
C	BM10392
C WRITE(25,714) (ACCWC(IRES),IRES=1,NRES)	BM10393
WRITE(26,714) (ACSHOP(ISHOP),ISHOP=1,NSHOP)	BM10394
714 FORMAT(10I8)	BM10395
713 CONTINUE	BM10396
DO 716 IRES=1,NRES	BM10397
716 ACUMRS(IRES)=0.	BM10398
DO 717 ISHOP=1,NSHOP	BM10399
717 ACUMSP(ISHOP)=0.	BM10400
IF(LAST.EQ.1) GOTO 61	BM10401
GOTO 41	BM10402
C	BM10403
C	BM10404
51	BM10405
I=CCDE	BM10406
CALL ADDL(1,PRI,IREADY,NREADY)	BM10407
IF(LAST.EQ.1) GOTO 81	BM10408
GO TO 41	BM10409
C	BM10410
C	BM10411
C	BM10412
ACTIVITY IS FINISHED .RELEASE RESOURCES AND UPDATE THE	BM10413
STATUS OF THE FOLLOWERS.	BM10414
C	BM10415
61 I=CCDE-2500	BM10416
PCUV=1.	BM10417
LENGTH=LENGTH+1	BM10418
IF(TIAS.LT.CLOCK) CALL ACCUM(TIAS,WCSHOP,NSHOP)	BM10419
C	BM10420
RELEASE RESOURCES	BM10421
NRS=ANUM(I)	BM10422
IF(NRS.EQ.0) GOTO 252	BM10423
DO 201 K=1,NRS	BM10424
L=WCSHOP(ASHOP(I,K))	BM10425
AR(L)=AR(L)+A(I,K)*2(I)	BM10426
201 CONTINUE	BM10427
252 CONTINUE	BM10428
C	BM10429
C	BM10430
C	BM10431
C	BM10432
C	BM10433
C	BM10434
C	BM10435
C	BM10436
C	BM10437
C	BM10438
C	BM10439
C	BM10440

	SET=SET+TATTR(IK,L+4)	BM10441
202	CONTINUE	BM10442
414	K=TFOLL(IK,L)	BM10443
	ITYRPR=L	BM10444
C	UPDATE THE STATUS OF THE FOLLOWER	BM10445
	STATUS(K)=STATUS(K)-1	BM10446
	IF(STATUS(K).GT.0) GOTO 241	BM10447
C	FIND THE TYPE OF THE ACTIVITY	BM10448
	IF(ITYPE(K).LE.2) GOTO 306	BM10449
	IF(ITYPE(K).EQ.5) GOTO 506	BM10450
	IF(ITYPE(K).EQ.9) GOTO 406	BM10451
306	CALL ADDL(K,PRI,IREADY,NREADY)	BM10452
	GOTO 241	BM10453
406	TSTART(K)=CLOCK	BM10454
	TFINIS(K)=CLOCK	BM10455
	CALL ADDL(K,PRI,IACTIV,NACTIV)	BM10456
	CALL SCHED(2500+K,CLOCK)	BM10457
	Z(K)=1.	BM10458
	GOTO 241	BM10459
506	ADUR=DUR(K)+CLOCK	BM10460
	CALL SCHED(2500+K,ADUR)	BM10461
	TSTART(K)=CLOCK	BM10462
	CALL ADDL(K,PRI,IACTIV,NACTIV)	BM10463
	TFINIS(K)=ADUR	BM10464
	Z(K)=1.	BM10465
241	CONTINUE	BM10466
C	UPDATE THE ATTRIBUTES OF TEST FOR NEXT OCCURANCE	BM10467
	TATTR(IK,1)=1.	BM10468
	DUR(I)=1	BM10469
C	FIND EXPECTED DURATIONS OF REPAIR AFTER TEST	BM10470
	DC 584 L=1,2	BM10471
	PRUB(L)=TATTR(IK,3+L)	BM10472
	K=TFOLL(IK,L)	BM10473
	RDUR(L)=0.	BM10474
	IF(K.EQ.0) GOTO 584	BM10475
	RDUR(L)=DUR(K)	BM10476
	NBA=TO(I)	BM10477
585	IS=FPOS(K)+1	BM10478
	NBK=TO(K)	BM10479
C	CHECK IF TO NODE OF FOLLOWER(K) IS EQUAL TO TO NODE OF TEST	BM10480
	IF(NBA.EQ.NBK)GO TO 584	BM10481
	RDUR(L)=PDUR(L)+DUR(K)	BM10482
	K=FOLL(IS)	BM10483
	GOTO 585	BM10484
584	CONTINUE	BM10485
	EDUR(I)=RDUR(ITYRPR)	BM10486
	TFINIS(I)=RDUR(ITYRPR)+CLOCK	BM10487
	GOTO 144	BM10488
212	CONTINUE	BM10489
C		BM10490
C	UPDATE THE STATUS OF THE FOLLOWER	BM10491
C		BM10492
	NPF=FPOS(I)+1	BM10493
	KPF=NF(I)+FPOS(I)	BM10494
	IF(NF(I).EQ.0) GOTO 144	BM10495

DC 24 L=NPF,KPF	BM104960
IF(CTRANF(L).NE.1) GOTO 24	BM104970
K=FCLL(L)	BM104980
STATUS(K)=STATUS(K)-1	BM104990
IF(STATUS(K).GT.0) GOTO 24	BM105000
C RELEASE FOLLOWER	BM105010
IF(ITYPE(K).LE.4) GOTO 30	BM105020
IF(ITYPE(K).EQ.5) GOTO 50	BM105030
IF(ITYPE(K).EQ.7) GOTO 45	BM105040
IF(ITYPE(K).EQ.9) GOTO 40	BM105050
WRITE(6,*)'ITYPE ERROR FOR ACT',K,'ITYPE(K)=',ITYPE(K)	BM105060
C	BM105070
C FOLLOWER IS A NOMAL ACTIVITY	BM105080
30 CCNTINUE	BM105090
IF(ISWTCH.EQ.1) THEN	BM105100
C DELAY THE FOLLOWER OF MILESTONE THAT HAS PREDEFINED DATE	BM105110
CALL SCHED(K,DMILE)	BM105120
GO TO 24	BM105130
ENDIF	BM105140
CALL ADDL(K,PRI,IREADY,NREADY)	BM105150
GOTO 24	BM105160
C DUMMY ACTIVITY	BM105170
40 TSTART(K)=CLOCK	BM105180
TFINIS(K)=CLOCK	BM105190
CALL ADDL(K,PRI,IACTIV,NACTIV)	BM105200
CALL SCHED(2500+K,CLOCK)	BM105210
Z(K)=1.	BM105220
GOTO 24	BM105230
C	BM105240
C ARRIVED AT A MILESTON-EVENT	BM105250
45 CCNTINUE	BM105260
TMILE(MILE(K))=CLOCK	BM105270
TSTART(K)=CLOCK	BM105280
TFINIS(K)=CLOCK	BM105290
DMILE=DUR(K)	BM105300
IF(TMILE(MILE(K)).LT.DMILE) ISWTCH=1	BM105310
GOTO 24	BM105320
C	BM105330
C "DELAY" ACTIVITY	BM105340
50 ADUR=DUR(K)+CLOCK	BM105350
TSTART(K)=CLOCK	BM105360
TFINIS(K)=ADUR	BM105370
Z(K)=1.	BM105380
CALL SCHED(2500+K,ADUR)	BM105390
CALL ADDL(K,PRI,IACTIV,NACTIV)	BM105400
24 CCNTINUE	BM105410
ISWTCH=0	BM105420
144 CCNTINUE	BM105430
C	BM105440
C UPDATE ACTIVE LIST	BM105450
CALL CANCEL(I,IACTIV,NACTIV)	BM105460
IF(LAST.EQ.1) GOTO 81	BM105470
GOTO 41	BM105480
C	BM105490
C FLOW-TRANSFERRED FOLLOWER CAN BE STARTED	BM105500

C		BM10551
71	I=CODE-5000	BM10552
	IF(NF(I).EQ.0) GOTO 302	BM10553
	NPF = FPOS(I)+1	BM10554
	KPF = FPOS(I)+NF(I)	BM10555
	DC 301 L=NPF,KPF	BM10556
	K=FCLL(L)	BM10557
	STATUS(K)=STATUS(K)-1	BM10558
	IF(STATUS(K).GT.0) GOTO 301	BM10559
	IF(CUR(K).LE.0.) GOTO 391	BM10560
	CALL ADDL(K,PRI,IREADY,NREADY)	BM10561
	GOTO 301	BM10562
391	CALL ADDL(K,PRI,IACTIV,NACTIV)	BM10563
	CALL SCHED(2500+K, CLOCK)	BM10564
	TSTART(K)=CLOCK	BM10565
	TFINIS(K)=CLOCK	BM10566
	Z(K)=ZU(K)	BM10567
301	CONTINUE	BM10568
302	IF(LAST.EQ.1) GOTO 81	BM10569
	GOTO 41	BM10570
C		BM10571
C	LAST EVENT AT CLOCK. ACCUMULATE RESOURCES USED	BM10572
C		BM10573
81	CALL ACCUM(TIAS,WCSHOP,NSHOP)	BM10574
	LAST=C	BM10575
C	CHECK FOR END OF SIMULATION	BM10576
C	MODIFIED NOT TO WRITE AVG. RES FOR THE LAST PERIOD...	BM10577
C	810 IF(NREADY.EQ.0.AND.NACTIV.EQ.0) GOTO 710	BM10578
C	810 IF(NREADY.EQ.0.AND.NACTIV.EQ.0) GOTO 41	BM10579
C		BM10580
881	CONTINUE	BM10581
	IF(NACTIV.EQ.0) GO TO 870	BM10582
C	CHECK THE ASSIGNMENT POLICY	BM10583
	IF(IRAPCL.NE.2) GO TO 870	BM10584
C		BM10585
C	UPGRADING AND DOWNGRADING	BM10586
C	-----	BM10587
C		BM10588
	DC 146 I=1,NACTIV	BM10589
	K=IACTIV(I)	BM10590
	IF(ITYPE(K).GT.2) GO TO 146	BM10591
	ZMEX=Z(K)-ZL(K)	BM10592
	IF(ZMEX.LE.0.) GO TO 146	BM10593
	ZNEW=ZL(K)	BM10594
C	ADD UP RELEASED RESOURCES	BM10595
	NRS=ANUM(K)	BM10596
	DC 148 L=1,NRS	BM10597
	J=WCSHOP(ASHOP(K,L))	BM10598
148	AR(J)=AR(J)+A(K,L)*ZMEX	BM10599
	CALL RESCHD(K,ZNEW,AR)	BM10600
146	CONTINUE	BM10601
C		BM10602
C	INTENSITY UPGRADING	BM10603
C	-----	BM10604
870	CONTINUE	BM10605

	NRUN=1	BP10606
882	CONTINUE	BP10607
	M=0	BP10608
	IF(NACTIV.EQ.0) GOTO 531	BP10609
C		BP10610
	DC 151 I=1,NACTIV	BP10611
	K=IACTIV(I)	BP10612
	IF(ITYPE(K).GT.4) GOTO 151	BP10613
	IF(Z(K).EQ.ZU(K)) GOTO 151	BP10614
	IF(NRUN.EQ.2) GOTO 190	BP10615
	IF(LS(K).GT.TFINIS(K)) GOTO 151	BP10616
190	CONTINUE	BP10617
	M=M+1	BP10618
	IREV(M)=K	BP10619
	DRV(M)=LS(K)-TFINIS(K)	BP10620
151	CONTINUE	BP10621
531	CONTINUE	BP10622
	MTURN=M	BP10623
	IF(NREADY.EQ.0) GOTO 532	BP10624
	DC 641 I=1,NREADY	BP10625
	M=M+1	BP10626
	J=IREADY(I)	BP10627
	IREV(M)=J	BP10628
641	DRV(M)=LS(J)-CLOCK-DUR(J)	BP10629
532	CONTINUE	BP10630
	IF(M.EQ.0) GOTO 200	BP10631
C		BP10632
C	SET PRIORITIES ACCORDING TO DRV(LATENESS SCORE)	BP10633
	CALL SORT(DRV,APRI,M,'NOWRITE')	BP10634
C		BP10635
	DC 161 I=1,M	BP10636
C		BP10637
	J1=APRI(I)	BP10638
	JK=IREV(J1)	BP10639
	ZMEX=ZU(JK)-Z(JK)	BP10640
	IF(DUR(JK).LE.0) GOTO 559	BP10641
	PCOV=PCV(JK)+(CLOCK-TLU(JK))*Z(JK)/DUR(JK)	BP10642
559	IF(DUR(JK).LE.0) PCOV=1.	BP10643
	IF(NRUN.EQ.2) GOTO 571	BP10644
	IF(LS(JK).LE.CLOCK) GOTO 571	BP10645
	ZC=((1.-PCOV)*DUR(JK))/(LS(JK)-CLOCK)-Z(JK)	BP10646
	IF(ZMEX.GT.ZC) ZMEX=ZC	BP10647
571	CONTINUE	BP10648
C		BP10649
	NRS=ANUM(JK)	BP10650
	IF(NRS.EQ.0) GOTO 96	BP10651
	DC 171 L=1,NRS	BP10652
	J=WCSHUP(ASHOP(JK,L))	BP10653
	IF(A(JK,L).LT..0001) GOTO 171	BP10654
	ZAR=AR(J)/A(JK,L)	BP10655
	IF(ZMEX.GT.ZAR) ZMEX=ZAR	BP10656
171	CONTINUE	BP10657
96	CONTINUE	BP10658
C		BP10659
	IF(ZMEX.LE.0.01) GOTO 161	BP10660

	ZNEW=ZMEX+Z(JK)	BM10661
C		BM10662
C	CHECK PREDECESSORS	BM10663
	IF(NP(JK).EQ.0) GOTO 310	BM10664
	ISTART=PPUS(JK)+1	BM10665
	IEND=NP(JK)+PPDS(JK)	BM10666
C		BM10667
C	CHECK FLOW TRANSFERS	BM10668
	DO 210 IPDS=ISTART,IEND	BM10669
	IF(CTRANP(IPDS).EQ.1) GOTO 210	BM10670
	IK=FRED(IPDS)	BM10671
	IF(TFINIS(IK).LE.CLOCK) GOTO 210	BM10672
	IF(DUR(JK).LE.0) GOTO 210	BM10673
	ZMAXI=DUR(JK)*(1.-CTRANP(IPDS)-PCLV)/(TFINIS(IK)-CLOCK)	BM10674
	IF(ZMAXI.LT.ZNEW) ZNEW=ZMAXI	BM10675
	ZMEX=ZNEW-Z(JK)	BM10676
210	CONTINUE	BM10677
310	CONTINUE	BM10678
C		BM10679
	IF(ZNEW.LT.ZL(JK)) GOTO 161	BM10680
C	UPDATE AVAILABLE RESOURCES	BM10681
	IF(NRS.EQ.0) GOTO 196	BM10682
	DO 172 K=1,NRS	BM10683
	L=WCSHOP(ASHOP(JK,K))	BM10684
	AR(L)=AR(L)-A(JK,K)*ZMEX	BM10685
172	CONTINUE	BM10686
196	CONTINUE	BM10687
C		BM10688
C	RESCHEDULE FINISH AND PROGRESS	BM10689
C		BM10690
	IF(JI.LE.MTURN) GOTO 115	BM10691
C	FOR NEWLY SCHEDULED ACTIVITIES	BM10692
	CALL CANCEL(JK,IREADY,NREADY)	BM10693
	CALL ADCL(JK,PRI,IACTIV,NACTIV)	BM10694
	TSTART(JK)=CLOCK	BM10695
	TLU(JK)=CLOCK	BM10696
115	CONTINUE	BM10697
C		BM10698
	IF(ZNEW.EQ.Z(JK)) GOTO 161	BM10699
	CALL RESCHD(JK,ZNEW,AR)	BM10700
161	CONTINUE	BM10701
C		BM10702
200	CONTINUE	BM10703
	IF (NRUN.EC.2) GOTO 41	BM10704
C		BM10705
C	RESCHEDULE TRULY SLACK ACT'S IF CRITICAL ACT. DELAYED	BM10706
C		BM10707
	CALL CRITIC(ZU,PRI)	BM10708
C		BM10709
	NRUN=2	BM10710
	GOTO 882	BM10711
C		BM10712
C	RESOURCE CAPACITY CHANGE	BM10713
C	-----	BM10714
222	CONTINUE	BM10715

K=CODE-7500	BMIC7160
DC 1121 I=1,NSHOP	BMIC7170
AR(I)=AR(I)+CIC(K,I)-CAP(I)	BMIC7180
CAP(I)=CIC(K,I)	BMIC7190
1121 CONTINUE	BMIC7200
C	BMIC7210
C SEARCH IF THERE IS ANY CAPACITY VIOLATION	BMIC7220
DC 1131 I=1,NSHOP	BMIC7230
IF(AR(I).LT.-0.01) GOTO 1141	BMIC7240
1131 CONTINUE	BMIC7250
1231 CONTINUE	BMIC7260
IF(LAST.EQ.1) GOTO 81	BMIC7270
GOTO 41	BMIC7280
C	BMIC7290
C DOWNGRADING INTENSITIES	BMIC7300
C	BMIC7310
C CREATE LIST OF ACTIVITIES TO DOWNGRADE	BMIC7320
1141 CONTINUE	BMIC7330
WRITE(6,*)'*** INTENSITY DOWNGRADING TRIED ***'	BMIC7340
M=0	BMIC7350
DU 1151 I=1,NACTIV	BMIC7360
K=IACTIV(I)	BMIC7370
IF(ITYPE(K).GT.4) GOTO 1151	BMIC7380
IF(Z(K).EQ.ZL(K)) GOTO 1151	BMIC7390
M=M+1	BMIC7400
IREV(M)=K	BMIC7410
DRV(M)=TFINIS(K)-LS(K)	BMIC7420
1151 CONTINUE	BMIC7430
IF(M.LE.0) GOTO 37	BMIC7440
CALL SORT(DRV,NPRI,M,'NOWRITE')	BMIC7450
ZLOW=1.	BMIC7460
IIRUN=1	BMIC7470
CALL DNGRD(M,ZLOW,Z,IIRUN,PRI)	BMIC7480
C	BMIC7490
IF(JFLAG.EQ.0) GOTO 1231	BMIC7500
C	BMIC7510
C INTERRUPT ACTIVITIES	BMIC7520
37 CONTINUE	BMIC7530
M=0	BMIC7540
DC 2152 I=1,NACTIV	BMIC7550
K=IACTIV(I)	BMIC7560
IF(ITYPE(K).GT.4) GOTO 2152	BMIC7570
IF(Z(K).LE..0001) GOTO 2152	BMIC7580
M=M+1	BMIC7590
IREV(M)=K	BMIC7600
DRV(M)=TFINIS(K)-LS(K)	BMIC7610
2152 CONTINUE	BMIC7620
IF(M.EQ.0) GOTO 1173	BMIC7630
CALL SORT(DRV,NPRI,M,'NOWRITE')	BMIC7640
ZLOW=0.	BMIC7650
IIRUN=2	BMIC7660
CALL DNGRD(M,ZLOW,Z,IIRUN,PRI)	BMIC7670
GOTO 1231	BMIC7680
C	BMIC7690
1173 CONTINUE	BMIC7700

C	THERE IS NO ACTIVITY AVAILABLE	BM10771C
	WRITE(6,*)'NO ACT AVAILABLE IN DOWNGRADING...LOGICAL ERROR..'	BM10772C
	STOP	BM10773C
C		BM10774C
C	END OF SIMULATION	BM10775C
C	-----	BM10776C
91	WRITE(6,*)'END OF SIMULATION. TIME IS ',CLOCK	BM10777C
C		BM10778C
C	WRITE OUTPUT FILES	BM10779C
	IF(NMILE.EQ.0) GO TO 95	BM10780C
	DO 99 I=1,NMILE	BM10781C
	II=TMILE(I)-1	BM10782C
	IF(II.EQ.-1) II=0	BM10783C
	CALL DCUNV(II,IMILE)	BM10784C
	SMILE(I)=IMILE(I)*10000+IMILE(2)*100+IMILE(3)	BM10785C
99	CONTINUE	BM10786C
	WRITE(57,714) (SMILE(I),I=1,NMILE)	BM10787C
	WRITE(27,714) (TMILE(I),I=1,NMILE)	BM10788C
C		BM10789C
95	NSHOPP=NSHOP+1	BM10790C
	ATOT(NSHOPP)=0.	BM10791C
	DO 5555 KKL=1,NSHOP	BM10792C
	ATOT(NSHOPP)=ATOT(NSHOPP)+TOTMHR(KKL)	BM10793C
5555	TCTMDY(KKL)=TOTMHR(KKL)	BM10794C
	TCTMDY(NSHOPP)=ATOT(NSHOPP)	BM10795C
	WRITE(28,714) (TCTMDY(I),I=1,NSHOPP)	BM10796C
C		BM10797C
	IF(1CYCLE.GT.1) GO TO 2002	BM10798C
	IF(IRAPCL.EQ.1) WRITE(36,2003) NAME,NCYCLE,ILOW,ILP	BM10799C
	IF(IRAPCL.EQ.2) WRITE(36,2004) NAME,NCYCLE,ILOW,ILP	BM10800C
	WRITE(36,2005) SMON,SDAY,SYR	BM10801C
	WRITE(36,2006)	BM10802C
C		BM10803C
	IF(ICALD.EQ.1) THEN	BM10804C
	DO 2000 I=1,NNODE	BM10805C
	KSTART=TSTART(I)+.5	BM10806C
	KFINIS=TFINIS(I)+.5-1	BM10807C
	IF(KFINIS.LT.KSTART) KFINIS=KSTART	BM10808C
	IF(ITYPE(I).EQ.7) THEN	BM10809C
	II=PMILE(I)	BM10810C
	KSTART=TMILE(II)-1	BM10811C
	IF(KSTART.EQ.-1) KSTART=0	BM10812C
	KFINIS=KSTART	BM10813C
	ENDIF	BM10814C
	IF(ITYPE(I).EQ.9) KSTART=KFINIS	BM10815C
	CALL DCUNV(KSTART,CALND1)	BM10816C
	CALL LCUNV(KFINIS,CALND2)	BM10817C
	WRITE(36,2001) I,FROM(I),TO(I),CALND1,CALND2	BM10818C
2000	CONTINUE	BM10819C
	ELSE	BM10820C
	DO 2009 I=1,NNODE	BM10821C
2009	WRITE(36,2008) I,FROM(I),TO(I),TSTART(I),TFINIS(I)	BM10822C
	ENDIF	BM10823C
2002	CONTINUE	BM10824C
C		BM10825C

```

C      WRITE RLN-STATUS REPORT
      IF(ICYCLE.EQ.1) THEN
        WRITE(40,1001) NAME
1001   FORMAT(A50)
        IF(IRAPOL.EQ.1) WRITE(40,1002)
        IF(IRAPOL.EQ.2) WRITE(40,1003)
        IF(NCYCLE.EQ.1) WRITE(40,1004)
        IF(NCYCLE.GT.1) WRITE(40,1005) NCYCLE
        IF(NTEST.EQ.0) WRITE(40,1006)
        IF(NTEST.EQ.1) WRITE(40,1007)
        WRITE(40,1009) NNODE,ITURN,NRES,NSHUP,ILUH,IUP,KTEND,NCC,NREPDT,
+       ISCFIN,NMILE
      ENDIF

1002 FORMAT('INTENSITY ASSIGNMENT POLICY : UPGRADING ONLY')
1003 FORMAT('INTENSITY ASSIGNMENT POLICY : UP & DOWNGRADING')
1004 FORMAT('NO SIMULATIONS : 1 CYCLE')
1005 FORMAT('SIMULATIONS : ',15,' CYCLES')
1006 FORMAT('TEST ACTIVITIES & REWORKS : NO')
1007 FORMAT('TEST ACTIVITIES & REWORKS : YES')
1009 FORMAT('NUMBER OF ACTIVITIES : ',15,/)
+       'NUM. OF ACT. (NO REWORKS) : ',15,/
+       'NUMBER OF WORKCENTERS : ',15,/
+       'NUMBER OF SHOPS : ',15,/
+       'LOWER BOUND OF INTENSITY : ',15,/
+       'UPPER BOUND OF INTENSITY : ',15,/
+       'LIMIT OF PROJECT FINISH : ',15,' (FOR SIMULATIONS)',/
+       'NUMBER OF CAPACITY CHANGE : ',15,/
+       'NUMBER OF REPORTING DATES : ',15,/
+       'PROJECT TARGET FINISH TIME : ',15,/
+       'NUMBER OF MILESTONES : ',15, )

C      REWIND 17
      WRITE(17,1777) NRAND

C
C
2003 FORMAT(1H1,120(1H=),//,6X,'PROJECT NAME : ',A20,15,' RUN(S)',
+ ' USING UPGRADING ONLY',
+ ' INTENSITY RANGE ',15,' - ',15, '//,12C('='))
2004 FORMAT(1H1,120(1H=),//,6X,'PROJECT NAME : ',A20,15,' RUN(S)',
+ ' USING UP & DOWN GRADING',
+ ' INTENSITY RANGE ',15,' - ',15, '//,12C('='))
2005 FORMAT(1H , ' PROJECT START DATE : ',315,/)
2006 FORMAT(' ACTIVITY',20X,' SCHEDULE',/,
+ ' INTERNAL #',3X,' I',5X,' J',4X,
+ ' START DATE',11X,' FINISH DATE',/,70('='))
2001 FORMAT(5X,I4,5X,A4,5X,A4,5X,3(1X,I2),13X,3(1X,I2))
2008 FORMAT(5X,I4,5X,A4,5X,A4,7X,F7.1,15X,F7.1)
323 FORMAT(A50,8(/49X,16))
1777 FORMAT(110)
951 FORMAT(20I4)
953 FORMAT(2CF4.2)
C 954 FORMAT(4F10.1)
952 FORMAT(6X,A4,2X,A4,3X,I4,2X,I1)
C 952 FORMAT(6X,A4,9X,I4,2X,I1)

```

BM108260
 BM108270
 BM108280
 BM108290
 BM108300
 BM108310
 BM108320
 BM108330
 BM108340
 BM108350
 BM108360
 BM108370
 BM108380
 BM108390
 BM108400
 BM108410
 BM108420
 BM108430
 BM108440
 BM108450
 BM108460
 BM108470
 BM108480
 BM108490
 BM108500
 BM108510
 BM108520
 BM108530
 BM108540
 BM108550
 BM108560
 BM108570
 BM108580
 BM108590
 BM108600
 BM108610
 BM108620
 BM108630
 BM108640
 BM108650
 BM108660
 BM108670
 BM108680
 BM108690
 BM108700
 BM108710
 BM108720
 BM108730
 BM108740
 BM108750
 BM108760
 BM108770
 BM108780
 BM108790
 BM108800

961	FORMAT(5X,415)	BM10681C
963	FORMAT(10F8.2)	BM10682C
3001	FORMAT(10F8.2)	BM10883C
1999	CONTINUE	BM10684C
9999	STOP	BM10885C
	END	BM10686C
		BM10887C
		BM10688C
		BM10889C
	SUBROUTINE CANCEL(I,IX,LX)	BM10690C
C	THIS SUBROUTINE CANCELS THE INDEX(I) FROM THE LIST (IX)	BM10691C
C	AND UPDATES THE LIST LENGTH (LX)	BM10892C
	DIMENSION IX(1000)	BM10693C
	IF(LX.EQ.0) GOTO 30	BM10894C
	NA=I	BM10895C
	DO 12 K=1,LX	BM10896C
	IF(NA.EQ.IX(K)) GO TO 22	BM10897C
12	CONTINUE	BM10898C
C	CODE "I" DOES NOT EXIT IN THE LIST.	BM10899C
	RETURN	BM10900C
22	LX=LX-1	BM10901C
	N=K	BM10902C
	DO 14 J=N,LX	BM10903C
	IX(J)=IX(J+1)	BM10904C
14	CONTINUE	BM10905C
	IX(LX+1)=0	BM10906C
30	CONTINUE	BM10907C
	RETURN	BM10908C
	END	BM10909C
		BM10910C
		BM10911C
	SUBROUTINE ADDL(I,PRI,IX,LX)	BM10912C
C	THIS SUBROUTINE ADDS INDEX (I) TO THE LIST (IX) AND UPDATES LIST	BM10913C
C	LENGTH, (LX)	BM10914C
	DIMENSION IX(1000), IY(1000)	BM10915C
	INTEGER PRI(2500)	BM10916C
	NA=I	BM10917C
	IF(LX.EQ.0) GOTO 30	BM10918C
	DO 12 K=1,LX	BM10919C
	JA=IX(K)	BM10920C
	IF(PRI(NA).LT.PRI(JA)) GOTO 20	BM10921C
12	CONTINUE	BM10922C
	LX=LX+1	BM10923C
	IX(LX)=NA	BM10924C
	GOTO 40	BM10925C
20	DO 14 J=K,LX	BM10926C
14	IY(J+1)=IX(J)	BM10927C
	IX(J)=NA	BM10928C
	LX=LX+1	BM10929C
	M=K+1	BM10930C
	DO 16 J=M,LX	BM10931C
16	IX(J)=IY(J)	BM10932C
	GOTO 40	BM10933C
		BM10934C
		BM10935C

```

30 IX(1)=NA
   LX=1
40 RETURN
   END

```

```

SUBROUTINE SORT(B1,P1,KLIMIT,WRTCOD)

```

```

C THIS SUBROUTINE SORTS A VECTOR B1, WHILE RESERVING ITS
C ORIGINAL ORDER IN VECTOR P1. THE SUBROUTINE USES THE
C "QUICKSORT" METHOD.

```

```

REAL B(4500),VECCUT(4500),B1(KLIMIT)
INTEGER P(4500), POUT(4500),P1(KLIMIT)
CHARACTER*7 WRTCOD
DO 100 I=1,KLIMIT
  B(I)=B1(I)

```

```

100 P(I)=I
    ALIMIT=FLOAT(KLIMIT)
    ANUM=ALCG10(ALIMIT)/ALOG10(2.)
    NUM=ANUM
    IF(ANUM.GT.NUM) NUM =NUM +1
    MM=2**NUM
    NRESID=MM-KLIMIT
    DO 101 L=1,NRESID
101  B(KLIMIT+L)=1.E10
    DO 6 I=1,NUM
      K=2**I
      M=MM/K
      DO 4 J1=1,M

```

```

        LSTART=K*(J1-1) +1
        LEND= LSTART+K-1
        K2=K/2
        K1=0

```

```

        J2=LSTART
1      IF(B(J2).LE.B(J2+K2)) GOTO 2
        VECCUT(LSTART+K1)=B(J2+K2)
        PCUT(LSTART+K1)=P(J2+K2)
        B(J2+K2)=1.E10
        K2=K2+1
        GOTO 3

```

```

2      PCUT(LSTART+K1)=P(J2)
        VECCUT(LSTART+K1)=B(J2)
        B(J2)=1.E10
        J2=J2+1
        K2=K2-1

```

```

3      J2=J2-LSTART+1
        IF(J3.GT.(K/2)) GOTO 31
        IF((J3+K2).GT.K) GOTO 31
        K1=K1+1
        GOTO 1

```

```

31  DO 32 L=LSTART,LEND
      IF(B(L).GE.1.E10) GOTO 32
      K1=K1+1
      VECCUT(LSTART+K1)=B(L)

```

```

BM109360
BM109370
BM109380
BM109390
BM109400
BM109410
BM109420
BM109430
BM109440
BM109450
BM109460
BM109470
BM109480
BM109490
BM109500
BM109510
BM109520
BM109530
BM109540
BM109550
BM109560
BM109570
BM109580
BM109590
BM109600
BM109610
BM109620
BM109630
BM109640
BM109650
BM109660
BM109670
BM109680
BM109690
BM109700
BM109710
BM109720
BM109730
BM109740
BM109750
BM109760
BM109770
BM109780
BM109790
BM109800
BM109810
BM109820
BM109830
BM109840
BM109850
BM109860
BM109870
BM109880
BM109890
BM109900

```

	POUT(LSTART+K1)=P(L)	BM109910
32	CONTINUE	BM109920
4	CONTINUE	BM109930
	DC 5 J=1,KLIMIT	BM109940
	P(J)=PCUT(J)	BM109950
5	B(J)=VECCUT(J)	BM109960
6	CONTINUE	BM109970
	DO 7 I=1,KLIMIT	BM109980
	IF(WRITCOD.NE.'NOWRITE') B1(I)=E(I)	BM109990
7	P1(P(I))=1	BM110000
	RETURN	BM110010
	END	BM110020
		BM110030
		BM110040
	SUBROUTINE PICK(P,K,KLIM)	BM110050
	DIMENSION P(100)	BM110060
	SET=0.	BM110070
	DC 20 I=1,KLIM	BM110080
	SET=SET+P(I)	BM110090
	CALL RAND(NRAND,URAND)	BM110100
	IF(URAND.LE.SET) GOTO 30	BM110110
20	CONTINUE	BM110120
30	K=1	BM110130
	RETURN	BM110140
	END	BM110150
		BM110160
		BM110170
	FUNCTION ENS(PS,ALFA)	BM110180
C	THIS FUNCTION CALCULATES EXPECTED NUMBER OF TRIALS NEEDED	BM110190
C	A TEST ACTIVITY TO BE SUCCESSFUL, PS=PROBABILITY OF SUCCESS	BM110200
C	AT THE CURRENT TRIAL AND ALFA CONSTANT INCREASE IN PROBABILITY	BM110210
C	OF SUCCESS AFTER EACH FAILURE.	BM110220
	ENS=0.	BM110230
	PROS=PS	BM110240
	PROF=1.	BM110250
	DO 10 I=1,20	BM110260
	AD=PROS*PROF*1	BM110270
	ENS =ENS+AD	BM110280
	IF(AD.LE.0.001) GOTO 20	BM110290
	PROF=PROF*(1.-PROS)	BM110300
	PROS=PROS*ALFA	BM110310
10	CONTINUE	BM110320
20	RETURN	BM110330
	END	BM110340
		BM110350
		BM110360
	REAL FUNCTION XMEAN(PROB,VAL,NUM)	BM110370
	DIMENSION PROB(10), VAL(10)	BM110380
C	THIS FUNCTION CALCULATES MEAN OF AN ARRAY(VAL), WITH	BM110390
C	PROBABILITIES (PROB), NUM BEING THE NUMBER OF ELEMENTS	BM110400
C	IN VAL	BM110410
	XMEAN=0.	BM110420
	DC 10 I=1,NUM	BM110430
10	XMEAN=XMEAN+PROB(I)*VAL(I)	BM110440
	RETURN	BM110450
	END	

```

SUBROUTINE ADJFIN(EDUR,ATFIN,NNODE)
DIMENSION PCV(2500), TLU(2500), TFINIS(2500),Z(2500)
DIMENSION ATFIN(2500),EDUR(2500)
INTEGER DUR(2500)
COMMON/BBB/ PCV,TLU,TFINIS,Z,CLOCK,DUR,LENGTH

```

C THIS SUBROUTINE FINDS ADJUSTED FINISH TIMES FOR TARGET FINISH
C CALCULATIONS.

```

      DO 10 I=1,NNODE
      IF(TFINIS(I).EQ.99999.) GOTO 11
      IF(TFINIS(I).LE.CLOCK) GOTO 11
      PCOV=PCV(I)+(CLOCK-TLU(I))*Z(I)/DUR(I)
      IF(Z(I).LE.0. .AND. PCOV.LE.0.) GOTO 11
      ATFIN(I)=CLOCK+EDUR(I)-PCOV*DUR(I)
      GOTO 10
11  ATFIN(I)=TFINIS(I)
10  CONTINUE
      RETURN
      END

```

```

SUBROUTINE LSCHED(DUR,TFINIS,ISCFIN,ITYPE)

```

C THIS SUBROUTINE FINDS EARLY & LATE FINISH TIMES FOR ACTIVITIES
C BY CPM

```

      INTEGER RANK(2500),ITYPE(2500)
      DIMENSION EF(2500), DUR(2500)
      REAL LS(2500),LFNA,TFINIS(2500)
      INTEGER FOLL(10000),NF(2500),FPOS(2500)
      INTEGER PRED(10000),NP(2500),PPOS(2500)
      DIMENSION CTRANF(10000),CTRANP(10000)
      COMMON/BB4/CTRANF,CTRANP
      COMMON/BB5/ LS,NNODE, RANK, ITURN
      COMMON/BB6/PRED, NP, PPOS
      COMMON/BB7/FOLL, NF, FPOS

```

C FORWARD ROUTINE

```

      FINIS=0.
      I=RANK(1)
      IF(TFINIS(I).NE.99999.) GOTO 50
      EF(I)=DUR(I)
      GOTO 60
50  EF(I)=TFINIS(I)
60  CONTINUE
      DO 100 K=2,ITURN
      NA=RANK(K)
      IF(TFINIS(NA).EQ.99999. ) GOTO 70
      EF(NA)=TFINIS(NA)
      GOTO 71
70  CONTINUE
      ESNA=0.
      IF(NP(NA).EQ.0) GOTO 201
      JP1=PPOS(NA)+1
      JP2=PPUS(NA)+NP(NA)

```

BP110460
BP110470
BP110480
BP110490
BP110500
BP110510
BP110520
BP110530
BP110540
BP110550
BP110560
BP110570
BP110580
BP110590
BP110600
BP110610
BP110620
BP110630
BP110640
BP110650
BP110660
BP110670
BP110680
BP110690
BP110700
BP110710
BP110720
BP110730
BP110740
BP110750
BP110760
BP110770
BP110780
BP110790
BP110800
BP110810
BP110820
BP110830
BP110840
BP110850
BP110860
BP110870
BP110880
BP110890
BP110900
BP110910
BP110920
BP110930
BP110940
BP110950
BP110960
BP110970
BP110980
BP110990
BP111000

DO 200 J=JP1,JP2	BM111010
EFDUR=DUR(NA)	BM111020
JA=FRED(J)	BM111030
IF(EFDUR.GT.DUR(JA)) EFDUR=DUR(JA)	BM111040
EFC=EF(JA)-(1.-CTRANP(J))*EFDUR	BM111050
ESNA=AMAX1(ESNA,EFC)	BM111060
200 CONTINUE	BM111070
201 CONTINUE	BM111080
EF(NA)=ESNA+DUR(NA)	BM111090
IF(ITYPE(NA).EQ.7) EF(NA)=ESNA	BM111100
71 CONTINUE	BM111110
FINIS=AMAX1(FINIS,EF(NA))	BM111120
100 CONTINUE	BM111130
C WRITE(24,12) FINIS	BM111140
C WRITE(24,494) (I,EF(I),I=1,NNODE)	BM111150
C 12 FORPAT(' PROJECT FINISH TIME = ',F8.2)	BM111160
C	BM111170
C BACKWARD ROUTINE	BM111180
C	BM111190
IF(ISCFIN.NE.0.AND.ISCFIN.LT.FINIS) THEN	BM111200
WRITE(6,*)'***** ERROR *****'	BM111210
WRITE(6,*)'SCHEDULE TARGET FINISH IS LESS THAN FORWARD FINISH'	BM111220
WRITE(6,*)'TARGET = ',ISCFIN,' FINIS = ',FINIS	BM111230
STOP	BM111240
ENDIF	BM111250
IF(ISCFIN.GT.0) FINIS=ISCFIN	BM111260
C	BM111270
NA=RANK(ITURN)	BM111280
LS(NA)=FINIS-DUR(NA)	BM111290
IF(ITYPE(NA).EQ.7) THEN	BM111300
LS(NA)=FINIS	BM111310
IF(LS(NA).GT.DUR(NA).AND.EF(NA).LT.DUR(NA)) LS(NA)=DUR(NA)	BM111320
IF(LS(NA).LT.DUR(NA)) THEN	BM111330
WRITE(6,*) ' ** MILESTONE DATE FOR NODE',NA	BM111340
WRITE(6,*) ' IS INACTIVE. PROGRAM WILL USE MCKE'	BM111350
WRITE(6,*) ' BINDING DATE AND CONTINUE.'	BM111360
DUR(NA)=0	BM111370
ENDIF	BM111380
ENDIF	BM111390
K=ITURN-1	BM111400
DO 300 JJ=1,K	BM111410
KK=ITURN-JJ	BM111420
NA=RANK(KK)	BM111430
LFNA=FINIS	BM111440
IF(LF(NA).EQ.0) GOTO 401	BM111450
JF1=FPOS(NA)+1	BM111460
JF2=FPOS(NA)+NF(NA)	BM111470
DO 400 J=JF1,JF2	BM111480
EFDUR=DUR(NA)	BM111490
JA=FOLL(J)	BM111500
IF(ITYPE(JA).EQ.7) EFDUR=0	BM111510
IF(EFDUR.GT.DUR(JA)) EFDUR=DUR(JA)	BM111520
LSC=LS(JA)+(1.-CTRANF(J))*EFDUR	BM111530
IF(LFNA.GE.LSC) LFNA=LSC	BM111540
400 CONTINUE	BM111550

401	CONTINUE	BM111560
	LS(NA)=LFNA-DUR(NA)	BM111570
C	SET MILESTONE DATE	BM111580
	IF(ITYPE(NA).EQ.7) THEN	BM111590
	LS(NA)=LFNA	BM111600
	IF(LS(NA).GT.DUR(NA).AND.EF(NA).LT.DUR(NA)) LS(NA)=DUR(NA)	BM111610
	IF(LS(NA).LT.DUR(NA)) THEN	BM111620
	WRITE(6,*) ' ** MILESTONE DATE FOR NODE',NA	BM111630
	WRITE(6,*) ' IS INACTIVE. PROGRAM WILL USE MORE'	BM111640
	WRITE(6,*) ' BINDING DATE AND CONTINUE.'	BM111650
	DUR(NA)=0	BM111660
	ENDIF	BM111670
	ENDIF	BM111680
300	CONTINUE	BM111690
	DO 80 I=1,NNODE	BM111700
	IF(I.LE.ITURN) GOTO 90	BM111710
	LS(I)=DUR(I)	BM111720
	GOTO 80	BM111730
90	LS(I)=LS(I)+DUR(I)	BM111740
	IF(ITYPE(I).EQ.7) LS(I)=LS(I)-DUR(I)	BM111750
80	CONTINUE	BM111760
C	WRITE(24,494) (I,LS(I),I=1,NNODE)	BM111770
C 494	FORMAT(5(18,F8.2))	BM111780
	RETURN	BM111790
	END	BM111800
	SUBROUTINE CHECK(XAR,IX)	BM111810
C		BM111820
C	THIS SUBROUTINE CHECKS IF CAPACITIES ARE VIOLATED.	BM111830
	DIMENSION XAR(200)	BM111840
	INTEGER WCSHOP(200),NSHOP	BM111850
	COMMON/CC9/WCSHOP,NSHOP	BM111860
	IX=0	BM111870
	DO 13 I=1,NSHOP	BM111880
	IF(XAR(I).LT.-.001) IX=1	BM111890
13	CONTINUE	BM111900
	RETURN	BM111910
	END	BM111920
		BM111930
		BM111940
		BM111950
	SUBROUTINE CRITIC(ZU,PRI)	BM111960
C		BM111970
C	THIS ROUTINE DELAYES "TRULY SLACK" ACTIVITIES TO GET RESOURCES	BM111980
C	ENOUGH TO START CRITICAL ACTIVITY THAT HAS DELAYED	BM111990
C		BM112000
	DIMENSION A(2500,70),AR(200),RNEED(200)	BM112010
	INTEGER ASHOP(2500,70),ANUM(2500)	BM112020
	DIMENSION ZL(2500),Z(2500),ZU(2500)	BM112030
	DIMENSION TSTART(2500),TFINIS(2500)	BM112040
	REAL LS(2500),PCV(2500),TLU(2500)	BM112050
	INTEGER WCSHOP(200),NSHOP	BM112060
	INTEGER PREL(10000),NP(2500),PPDS(2500)	BM112070
	REAL CTRANP(10000),CTRANP(10000)	BM112080
	INTEGER PRI(2500),DUR(2500),RANK(2500)	BM112090
		BM112100

	INTEGER IACTIV(1000), IREADY(1000)	BM112110
	INTEGER NSAVE(900), MSAVE(900), IRESCH(900)	BM112120
	COMMON/BB1/ IREADY, IACTIV, NREADY, NACTIV	BM112130
	COMMON/BB4/ CTRANF, CTRANP	BM112140
	COMMON/BB5/ LS, NNODE, RANK, ITURN	BM112150
	COMMON/BB6/ PRED, NP, PPOS	BM112160
	COMMON/BB8/ PCV, TLU, TFINIS, Z, CLOCK, DUR, LENGTH	BM112170
	COMMON/BB9/ ANUM, ASHOP, A, AR	BM112180
	COMMON/BB11/ ZL	BM112190
	COMMON/BB13/ NRES, NCC	BM112200
	COMMON/CC9/ WCSHOP, NSHOP	BM112210
C	ISAVE=0	BM112220
	JSAVE=0	BM112230
C		BM112240
	DO 10 II=1, NREADY	BM112250
C		BM112260
	SEARCH CRITICAL ACTIVITY	BM112270
C	I=IREADY(II)	BM112280
	IF((LS(I)-CLOCK-DUR(I)).GT.0) GO TO 10	BM112290
	MRES=ANUM(I)	BM112300
	IF(MRES.EQ.0) GO TO 10	BM112310
C		BM112320
	WRITE(37,100) CLOCK,I	BM112330
100	FORMAT(1X,F6.2,' ACT.',I4,' IN CRITICAL ROUTINE *****')	BM112340
C		BM112350
	SET RESOURCES NEEDED	BM112360
C	DO 12 L=1, MRES	BM112370
	K=WCSHOP(ASHOP(I,L))	BM112380
12	RNEED(K)=A(I,L)*ZL(I)-AR(K)	BM112390
C		BM112400
	DO 13 L=1, 900	BM112410
13	IRESCH(L)=0	BM112420
C		BM112430
	SEARCH TRULY SLACK ACTIVITIES	BM112440
C	IDX=0	BM112450
	DO 15 JJ=1, NACTIV	BM112460
	J=IACTIV(JJ)	BM112470
	IF((LS(J)-CLOCK-DUR(J)).LE.0) GO TO 15	BM112480
	IF(ANUM(J).LE.0) GO TO 15	BM112490
	IDX=IDX+1	BM112500
	IRESCH(IDX)=J	BM112510
15	CONTINUE	BM112520
C		BM112530
	IF(IRESCH(1).EQ.0) RETURN	BM112540
C		BM112550
	CHECK IF ENOUGH RESOURCES CAN BE ACQUIRED	BM112560
C	NN=1	BM112570
	J=IRESCH(NN)	BM112580
16	NCT ENOUGH RES. FAIL	BM112590
C	IF(J.EQ.0) THEN	BM112600
	GO TO 10	BM112610
	ENDIF	BM112620
	MRES=ANUM(J)	BM112630
	DO 17 L=1, MRES	BM112640
		BM112650

	K=WCSHOP(ASHOP(J,L))	BMI12660
17	RNEED(K)=RNEED(K)-A(J,L)*Z(J)	BMI12670
	MRES=ANUM(I)	BPI12680
	ICHECK=0	BPI12690
	DC 18 L=1,MRES	BMI12700
	K=WCSHOP(ASHOP(I,L))	BMI12710
18	IF(RNEED(K).GT.0) ICHECK=1	BMI12720
C	IF ICHECK=1, TRY MORE SLACK ACTIVITIES	BPI12730
	IF(ICHECK.EQ.1) THEN	BPI12740
	NN=NN+1	BMI12750
	GC TO 16	BPI12760
	ENDIF	BPI12770
C		BPI12780
C	SUCCESS TO GET ENOUGH RESOURCES	BMI12790
C	RELEASE RESOURCES OF NN SLACK ACT'S	BPI12800
	DC 20 M=1,NN	BPI12810
	J=IRESCH(M)	BPI12820
	MRES=ANUM(J)	BMI12830
	DC 20 L=1,MRES	BPI12840
	K=WCSHOP(ASHOP(J,L))	BPI12850
	AR(K)=AR(K)+A(J,L)*Z(J)	BPI12860
20	CONTINUE	BMI12870
C		BPI12880
C	SCHEDULE ACTIVITY I	BPI12890
	ZNEW=ZU(I)	BMI12900
	ZC=CUR(I)/(LS(I)-CLOCK)	BPI12910
	IF(ZNEW.GT.ZC) ZNEW=ZC	BMI12920
	MRES=ANUM(I)	BPI12930
	DC 27 L=1,MRES	BPI12940
	K=WCSHOP(ASHOP(I,L))	BPI12950
	IF(A(I,L).LT..0001) GOTO 27	BPI12960
	ZAR=AR(K)/A(I,L)	BMI12970
	IF(ZNEW.GT.ZAR) ZNEW=ZAR	BPI12980
27	CONTINUE	BMI12990
C	CHECK FLOW TRANSFER	BPI13000
	IF(NP(I).EQ.0) GOTO 31	BMI13010
	ISTART=PPOS(I)+1	BPI13020
	IEND=NP(I)+PPQS(I)	BPI13030
	DC 29 IPCS=ISTART,IEND	BPI13040
	IF(CTRANP(IPQS).EQ.1) GOTO 29	BMI13050
	IK=PRED(IPCS)	BPI13060
	IF(TFINIS(IK).LE.CLOCK) GOTO 29	BMI13070
	IF(CUR(I).LE.0) GOTO 29	BPI13080
	ZMAXI=CUR(I)/(1.-CTRANP(IPQS))/(TFINIS(IK)-CLOCK)	BPI13090
	IF(ZMAXI.LT.ZNEW) ZNEW=ZMAXI	BPI13100
29	CONTINUE	BPI13110
31	CONTINUE	BMI13120
C		BPI13130
	IF(ZNEW.LT.ZL(I)) THEN	BPI13140
C	CLEAR THE ADDED RESOURCES	BPI13150
	DC 32 P=1,NN	BPI13160
	J=IRESCH(P)	BPI13170
	MRES=ANUM(J)	BPI13180
	DC 32 L=1,MRES	BPI13190
	K=WCSHLP(ASHOP(J,L))	BPI13200

32	AR(K)=AR(K)-A(J,L)*Z(J)	BM11321C
	CONTINUE	BM11322C
	GO TO 10	BM11323C
	ENDIF	BM11324C
C		BM11325C
C	UPDATE AVAILABLE RESOURCES,ETC. AND SAVE I IN NSAVE	BM11326C
	DO 33 L=1,PRES	BM11327C
	K=WCSHOP(ASHOP(I,L))	BM11328C
	AR(K)=AR(K)-A(I,L)*ZNEW	BM11329C
33	CONTINUE	BM11330C
	ISAVE=ISAVE+1	BM11331C
	NSAVE(ISAVE)=I	BM11332C
	TSTART(I)=CLOCK	BM11333C
	TLU(I)=CLOCK	BM11334C
	CALL RESCHD(I,ZNEW,AR)	BM11335C
	WRITE(37,106) I	BM11336C
106	FORMAT(' ACTIVITY',I5,' IS SCHEDULED.')	BM11337C
C		BM11338C
C	UPDATE UNDOED ACTIVITIES AND SAVE J INTO MSAVE	BM11339C
	DO 35 M=1,NN	BM11340C
	J=IKESCH(M)	BM11341C
	WRITE(37,107) CLOCK,J,CLOCK+1.	BM11342C
107	FORMAT(1X,F6.2,' ACT.',I4,' IS DELAYED UNTIL',F6.2)	BM11343C
	Z(J)=0.	BM11344C
	PCV(J)=0.	BM11345C
	CALL CANCEL(J,IACTIV,NACTIV)	BM11346C
	MSAVE(JSAVE+M)=J	BM11347C
35	CONTINUE	BM11348C
	JSAVE=JSAVE+NN	BM11349C
C		BM11350C
10	CONTINUE	BM11351C
C		BM11352C
C	UPDATE ACTIVE AND READY LIST	BM11353C
	IF(ISAVE.EQ.0) RETURN	BM11354C
	DO 50 II=1,ISAVE	BM11355C
	I=NSAVE(II)	BM11356C
	CALL CANCEL(I,IREADY,NREADY)	BM11357C
	CALL ADDL(I,PRI,IACTIV,NACTIV)	BM11358C
50	CONTINUE	BM11359C
	IF(JSAVE.EQ.0) THEN	BM11360C
	WRITE(*,*)'**** LOGIC ERROR IN SUBROUTINE CRITIC.. CHECK ****'	BM11361C
	RETURN	BM11362C
	ENDIF	BM11363C
	DO 60 JJ=1,JSAVE	BM11364C
	J=MSAVE(JJ)	BM11365C
	CALL ADDL(J,PRI,IREADY,NREADY)	BM11366C
6C	CONTINUE	BM11367C
C		BM11368C
	RETURN	BM11369C
	END	BM11370C
		BM11371C
		BM11372C
C	SUBROUTINE RESCHD(NA,ZNEW,AR)	BM11373C
C		BM11374C
C	THIS ROUTINE RESETS TFINIS,Z,PCOV,TLU ACCORDING TO NEW INTENSITY	BM11375C

C	DIMENSION DFL(30),PCV(2500),TLU(2500),TFINIS(2500),Z(2500)	BM113760
	INTEGER DUR(2500)	BM113770
	INTEGER FOLL(10000),NF(2500),FPDS(2500)	BM113780
	INTEGER PRED(10000),NP(2500),PPDS(2500)	BM113790
	DIMENSION TEVENT(10000)	BM113800
	INTEGER TIMVEC(10000,2),ADTIME(1000,11)	BM113810
	DIMENSION CTRANF(10000),CTRANP(10000),AR(200)	BM113820
	COMMON/BB4/CTRANF,CTRANP	BM113830
	COMMON/BB6/PRED,NP,PPDS	BM113840
	COMMON/BB7/FOLL,NF,FPDS	BM113850
	COMMON/BB8/PCV,TLU,TFINIS,Z,CLOCK,DUR,LENGTH	BM113860
	COMMON/111/TEVENT,TIMVEC,ADTIME,NXTVEC	BM113870
	PCOV=1.	BM113880
	ZOLD=Z(NA)	BM113890
	IF(DUR(NA).LE.0) GOTO 1254	BM113900
	PCOV=PCV(NA)+(CLOCK-TLU(NA))*Z(NA)/DUR(NA)	BM113910
1254	K=1	BM113920
	NPF=FPDS(NA)+1	BM113930
	CFL(1)=1.	BM113940
	IF(CTRANF(NPF).EQ.1.) GOTO 115	BM113950
	K=2	BM113960
	DFL(2)=CTRANF(NPF)	BM113970
115	CONTINUE	BM113980
C	ERASE OLD EVENT FROM SCHED LIST	BM113990
	IF(TEVENT(5000+NA).EQ.0.) GOTO 201	BM114000
	IF((DFL(2)-PCOV).LE.0.0001) GOTO 201	BM114010
	CALL ERASE(5000+NA)	BM114020
201	IF(TEVENT(2500+NA).EQ.0.) GOTO 202	BM114030
	IF(PCOV.GE.0.9999) GO TO 202	BM114040
	CALL ERASE(2500+NA)	BM114050
C	RESCHEDULE NEW EVENTS	BM114060
202	PCV(NA)=PCOV	BM114070
	TLU(NA)=CLOCK	BM114080
	IF(ZNEW.LE. .0001) THEN	BM114090
	TFINIS(NA)=99999.	BM114100
	WRITE(6,*)'ACT',NA,' INTERRUPTED AT',CLOCK	BM114110
	GO TO 26	BM114120
	ENDIF	BM114130
	DO 120 J=1,K	BM114140
	PLEFT=DFL(J)-PCOV	BM114150
	IF(PLEFT.LE.0.0001) GOTO 120	BM114160
	ANWTM=CLOCK+PLEFT*DUR(NA)/ZNEW	BM114170
	IF(DFL(J).LT.1.)GOTO 122	BM114180
	CALL SCHED(2500+NA,ANWTM)	BM114190
121	TFINIS(NA)=ANWTM	BM114200
	GO TO 120	BM114210
122	CONTINUE	BM114220
	CALL SCHED(5000+NA,ANWTM)	BM114230
120	CONTINUE	BM114240
26	CONTINUE	BM114250
	Z(NA)=ZNEW	BM114260
C	WRITE(37,148) NA,Z(NA),PCOV,CLOCK,ZOLD	BM114270
C	WRITE(37,148) CLECK,NA,Z(NA),ZOLD,PCOV,TFINIS(NA)	BM114280
		BM114290
		BM114300

C	4	,AR(01),AR(02)	BM114310
		LENGTH=LENGTH+1	BM114320
C	148	FORMAT(5X,15,5X,F6.3,5X,F6.3,5X,F8.2,5X,F6.3)	BM114330
	148	FORMAT(1X,F6.2,' ',14,' 1=',F6.3,	BM114340
	+	' 0=',F6.3,' P=',F5.3,' F=',F6.2,	BM114350
	+	' R=',F8.1)	BM114360
		RETURN	BM114370
		END	BM114380
		SUBROUTINE DWNGRD(M,ZLOW,Z,IIRUN,PRI)	BM114390
C			BM114400
C		THIS ROUTINE DOWNGRADES ACTIVITY INTENSITIES TO RECTIFY CAPACITY	BM114410
C		VIOLATIONS.	BM114420
		DIMENSION IREV(1000),NPRI(1000),A(2500,70),AR(200)	BM114430
		INTEGER ANUM(2500),ASHOP(2500,70),WCSHOP(200),NSHCP	BM114440
		INTEGER IACTIV(1000),IREADY(1000)	BM114450
		DIMENSION ZL(2500),Z(2500)	BM114460
		INTEGER PRI(2500)	BM114470
		COMMON/BB1/ IREADY, IACTIV,NREADY,NACTIV	BM114480
		COMMON/CC9/WCSHOP,NSHOP	BM114490
		COMMON/BB9/ANUM,ASHOP,A,AR	BM114500
		COMMON/BB10/IREV,NPRI,JFLAG	BM114510
		COMMON/BB11/ZL	BM114520
			BM114530
			BM114540
		DO 1162 I=1,M	BM114550
		J=NPRI(I)	BM114560
		JK=IREV(J)	BM114570
		IF(ZLOW.GT.ZL(JK)) ZLOW=ZL(JK)	BM114580
		ZMEX=Z(JK)-ZLOW	BM114590
C		DETERMINE IF ANY CONTRIBUTION DONE DUE TO JK	BM114600
		IFLAG=0	BM114610
		JFLAG=0	BM114620
		NRS=ANUM(JK)	BM114630
		IF(NRS.EQ.0) GOTO 1162	BM114640
		DO 1163 L=1,NRS	BM114650
		K=WCSHOP(ASHOP(JK,L))	BM114660
		AD=ZMEX+A(JK,L)	BM114670
		IF(AD.GT. .01 .AND. AR(K).LT.0.) IFLAG=1	BM114680
	1163	CONTINUE	BM114690
		IF(IFLAG.EQ.0) GOTO 1162	BM114700
C		UPDATE AVAILABLE RESOURCES	BM114710
		DO 1246 L=1,NRS	BM114720
		K=WCSHOP(ASHOP(JK,L))	BM114730
	1246	AR(K)=AR(K)+ZMEX+A(JK,L)	BM114740
		CALL RESCHD (JK,ZLOW,AR)	BM114750
C		CHECK IF CAPACITY VIOLATION RECTIFIED	BM114760
	1233	CALL CHECK(AR,JFLAG)	BM114770
		IF(JFLAG.EQ.0) RETURN	BM114780
	1162	CONTINUE	BM114790
C			BM114800
C		NEGATIVE AR =] LOGICAL ERROR	BM114810
		IF(IIRUN.EQ.2) THEN	BM114820
		WRITE(6,*)'NEGATIVE AR AFTER DISRUPTING .. LOGIC ERROR'	BM114830
		DO 1222 II=1,NSHOP	BM114840
		IF(AR(II).LT.0) THEN	BM114850

WRITE(6,*)'I,AR=',II,AR(II)	BM114860
ENDIF	BM114870
1222 CCNTINUE	BM114880
STOP	BM114890
ENDIF	BM114900
RETURN	BM114910
C	BM114920
END	BM114930
	BM114940
SUBROUTINE RAND(N,U)	BM114950
C	BM114960
C THIS ROUTINE COMPUTES A RANDOM NUMBER $0 < 'U' < 1$ USING SEED 'N'	BM114970
C	BM114980
N=N*843314861 + 453816693	BM114990
IF(N.GE.0) GOTO 1	BP115000
N=N + 2147483647 +1	BM115010
1 U=N* .4656612E-9	BP115020
IF(U.LT.0. OR.U.GT.1) WRITE(6,*)' RAND OUT OF LIMIT',U	BP115030
RETURN	BM115040
END	BM115050
	BP115060
	BM115070
	BM115080
SUBROUTINE SCHED(CODE,TIME)	BP115090
C	BM115100
C THIS SUBROUTINE ADDS THE EVENT TO THE SCHEDULING LIST.	BM115110
C	BP115120
INTEGER TIMVEC(10000,2),ADTIME(1000,11),NPCELL,NP2,CCDE,LTIME	BM115130
REAL TIME,TEVENT(10000)	BP115140
COMMON/TTT/TEVENT,TIMVEC,ADTIME,NXTVEC	BM115150
COMMON/YYY/LTIME	BP115160
	BM115170
NPCELL=TIME*10+1+.5	BP115180
IF(NPCELL.GT.LTIME) LTIME=NPCELL	BP115190
IF(TIMVEC(NPCELL,1).NE.0) GOTO 1	BM115200
C SCHEDULE FIRST EVENT AT 'TIME'	BP115210
TIMVEC(NPCELL,1)=CCDE	BM115220
GOTO 7	BP115230
C	BM115240
1 IF(TIMVEC(NPCELL,2).NE.0) GOTO 2	BP115250
C FIRST USE OF AUX. STORAGE AT 'TIME'	BP115260
ADTIME(NXTVEC,1)=CODE	BP115270
TIMVEC(NPCELL,2)=NXTVEC	BP115280
GOTO 6	BP115290
2 NP2=TIMVEC(NPCELL,2)	BP115300
3 IF(ADTIME(NP2,10).NE.0) GOTO 5	BP115310
C AUX. VECTOR DOESN'T POINT TO A NEXT ONE.	BP115320
DO 4 I=1,9	BP115330
IF(ADTIME(NP2,I).NE.0) GOTO 4	BP115340
ADTIME(NP2,I)=CCDE	BP115350
GOTO 7	BP115360
4 CONTINUE	BP115370
C EXTENSION VECTOR IS FULL. USE THE NEXT ONE AVAILABLE	BP115380
ADTIME(NP2,10)=NXTVEC	BP115390
ADTIME(NXTVEC,1)=CCDE	BP115400

C	FILL BACKWARD POINTER	BM115410
	ADTIME(NXTVEC,11)=NP2	BP115420
	GOTO 6	BP115430
5	NP2=ADTIME(NP2,10)	BP115440
	GOTO 3	BP115450
6	NXT1=NXTVEC+1	BP115460
C	FIND NEXT AVAILABLE AUX VECTOR	BP115470
	DO 8 I=NXT1,1000	BP115480
	IF(ADTIME(I,1).NE.0) GOTO 8	BP115490
	NXTVEC=I	BP115500
7	TEVENT(CCODE)=TIME	BP115510
	RETURN	BP115520
8	CONTINUE	BP115530
	WRITE (6,*) 'AUXILARRY EVENT STORAGE IS FULL. '	BP115540
	STOP	BP115550
	END	BP115560
		BP115570
		BP115580
	SUBROUTINE RMV(CODE,TIME,LAST,CLOCK)	BP115590
C	-----	BP115600
C	THIS SUBROUTINE FINDS THE THE NEXT EVENT IN THE LIST AND	BP115610
C	ERASES IT. IF THERE ARE MORE THEN A SINGLE EVENT AT TIME,	BP115620
C	RMV WILL PICK THE LAST ONE WHICH WAS SCHEDULED.	BP115630
C	-----	BP115640
C		BP115650
	INTEGER FCELL,TIMVEC(10000,2),ADTIME(1000,11),CCODE,LAST,LTIME	BP115660
	REAL TIME,CLOCK,STOPTM,TEVENT(10000)	BP115670
	COMMON/RMVMN/TEND	BP115680
	COMMON/TTT/TEVENT,TIMVEC,ADTIME,NXTVEC	BP115690
	COMMON/YYY/LTIME	BP115700
	LAST=0	BP115710
	FCELL=CLOCK*10+1+.5	BP115720
	DO 10 I=FCELL,LTIME	BP115730
	IF(TIMVEC(I,1).EQ.0) GOTO 10	BP115740
	IF(TIMVEC(I,2).NE.0) GOTO 1	BP115750
C	SINGLE EVENT AT 'TIME'	BP115760
	CODE=TIMVEC(I,1)	BP115770
C	TIME=FLCAT(I-1)/10. MODIFIED TO USE EXACT TIME VALUE	BP115780
	TIME=TEVENT(CCODE)	BP115790
	TIMVEC(I,1)=0	BP115800
	LAST=1	BP115810
	GOTO 32	BP115820
C		BP115830
C	MORE THEN ONE EVENT AT 'TIME'	BP115840
1	NP2=TIMVEC(I,2)	BP115850
2	IF(ADTIME(NP2,10).NE.0) GO TO 5	BP115860
	DO 4 J=2,10	BP115870
	IF(ADTIME(NP2,J).NE.0) GOTO 4	BP115880
	CCODE = ADTIME(NP2,J-1)	BP115890
C	TIME=(I-1)/10.	BP115900
	TIME=TEVENT(CODE)	BP115910
	ADTIME(NP2,J-1)=0	BP115920
C	ERASE THE THE LAST EVENT AT 'TIME'. IF IT IS THE ONLY ONE	BP115930
C	IN THE AUXILLARY VECTOR, RELEASE THE VECTOR.	BP115940
		BP115950

IF(J.NE.2) GOTO 32	BM115960
NP2=ADTIME(NP2,11)	BM115970
IF(NP2.NE.0) GOTO 3	BM115980
TIMVEC(1,2)=0	BM115990
GOTO 31	BM116000
3 ADTIME(NP2,11)=0	BM116010
ADTIME(NP2,10)=0	BM116020
31 IF(NXTVEC.GT.NP2) NXTVEC=NP2	BM116030
32 IF(TIME.GE.TEND) GOTO 11	BM116040
RETURN	BM116050
4 CONTINUE	BM116060
5 NP2=ADTIME(NP2,10)	BM116070
GOTO 2	BM116080
10 CONTINUE	BM116090
WRITE(6,*) 'SCHEDULER TABLE IS EMPTY'	BM116100
LAST=2	BM116110
RETURN	BM116120
11 WRITE(6,*) 'TIME LIMIT EXCEEDED. TIME IS ',TIME,' DAYS'	BM116130
WRITE(6,*) 'CODE=',CODE	BM116140
LAST=3	BM116150
END	BM116160
C	BM116170
SUBROUTINE ERASE(CODE)	BM116180
C	BM116190
C-----	BM116200
C THIS SUBROUTINE IS ERASING EVENTS FROM THE SCHEDULER LIST	BM116210
C-----	BM116220
C	BM116230
INTEGER TIMVEC(10000,2),ADTIME(1000,11),CODE,CODE1	BM116240
REAL TIME,TEVENT(10000)	BM116250
COMMON/TTT/TEVENT,TIMVEC,ADTIME,NXTVEC	BM116260
	BM116270
TIME=TEVENT(CODE)	BM116280
IF(TIME.EQ.0.) RETURN	BM116290
C *** EVENT HADN'T BEEN SCHEDULED	BM116300
TEVENT(CODE)=0.	BM116310
NCELL=TIME*10+1 +.5	BM116320
C THE TRANSFORMATION IS AS FOLLOWS	BM116330
C TIME*10 +1 (FOR T=0.) +.5 (FOR ROUNDING.)	BM116340
IF(TIMVEC(NCELL,1).NE.CODE) GOTO 1	BM116350
CALL RMV(CODE1,TIME,L,TIME)	BM116360
IF(CODE.EQ.CODE1) RETURN	BM116370
TIMVEC(NCELL,1)=CODE1	BM116380
RETURN	BM116390
1 NP2=TIMVEC(NCELL,2)	BM116400
2 DO 3 J=1,9	BM116410
IF(ADTIME(NP2,J).NE.CODE) GOTO 3	BM116420
CALL RMV(CODE1,TIME,L,TIME)	BM116430
IF(CODE1.EQ.CODE) RETURN	BM116440
ADTIME(NP2,J)=CODE1	BM116450
3 CONTINUE	BM116460
NP2=ADTIME(NP2,10)	BM116470
IF(NP2.GT.C) GOTO 2	BM116480
RETURN	BM116490
END	BM116500

```

SUBROUTINE ACCUM(TIAS,WCSHOP,NSHOP)
-----
C THIS SUBROUTINE ACCUMULATES THE USAGE OF THE RESOURCES.
C -----
C   INTEGER WCSHOP(200),NSHOP,ANUM(2500),ASHOP(2500,7C),IACTIV(1000)
C   INTEGER IREADY(1000),DUR(2500)
C   REAL ACUMRS(200),ACUMSP(50),A(2500,7C),CLOCK,TIAS,Z(2500)
C   REAL PCV(2500),TLU(2500),TFINIS(2500),AR(200)
C   COMMON/BB1/IREADY,IACTIV,NREADY,NACTIV
C   COMMON/BB8/PCV,TLU,TFINIS,Z,CLOCK,DUR,LENGTH
C   COMMON/BB9/ANUM,ASHOP,A,AR
C   COMMON/CC10/ACUMRS,ACUMSP
C
C   IF(TIAS.FQ.CLOCK) RETURN
C   DO 1 I=1,NACTIV
C     II=IACTIV(I)
C     NRS=ANUM(II)
C     IF(NRS.EQ.0) GOTO 1
C     DO 2 J=1,NRS
C       DRES=A(II,J)*(CLOCK-TIAS)*Z(II)
C       L=ASHOP(II,J)
C       ACUMRS(L)=ACUMRS(L)+DRES
C       ACUMSP(WCSHOP(L))=ACUMSP(WCSHOP(L))+DRES
C     2 CONTINUE
C   1 CONTINUE
C   TIAS=CLOCK
C   RETURN
C   END
BM116510
BM116520
BM116530
BM116540
BM116550
BM116560
BM116570
BM116580
BM116590
BM116600
BM116610
BM116620
BM116630
BM116640
BM116650
BM116660
BM116670
BM116680
BM116690
BM116700
BM116710
BM116720
BM116730
BM116740
BM116750
BM116760
BM116770
BM116780
BM116790
BM116800
BM116810
BM116820
BM116830
BM116840
BM116850
BM116860
BM116870
BM116880
BM116890
BM116900
BM116910
BM116920
BM116930
BM116940
BM116950
BM116960
BM116970
BM116980
BM116990
BM117000
BM117010
BM117020
BM117030
BM117040
BM117050

SUBROUTINE DCONV(WKD,CLND)
C *****
C *
C * THIS ROUTINE CONVERTS THE WORKING DAY INTO CALENDAR DATE
C *
C *****
C   INTEGER WKD,YEAR,SMON,SDAY,SYR,WKDTB(120,31),CLND(3)
C   COMMON/DCON/SMON,SDAY,SYR,YEAR,WKDTB
C
C   IF(WKD.LT.0) THEN
C     WRITE(6,*) '≠ ILLEGAL WORKING DATE =',WKD
C     CLND(1)=0
C     CLND(2)=0
C     CLND(3)=0
C     RETURN
C   ENDIF
C
C   IF(WKD.GE.99998) THEN
C     CLND(1)=99
C     CLND(2)=99
C     CLND(3)=99
C     RETURN
C   ENDIF

```

ISYR=(SYR-YEAR)/12+SMON	BM117060
M=WKD+1	BM117070
DC 310 JI=SDAY,31	BM117080
M=M-WKDTB(ISYR,JI)	BM117090
JOUT=JI	BM117100
IF(M.EQ.0)GOTO 380	BM117110
310 CONTINUE	BM117120
JSYR=ISYR+1	BM117130
320 DC 330 JJ=1,31	BM117140
M=M-WKDTB(JSYR,JJ)	BM117150
JOUT=JJ	BM117160
IF(M.EQ.0)GOTO 390	BM117170
330 CCNTINUE	BM117180
JSYR=JSYR+1	BM117190
GOTO 320	BM117200
380 CLND(1)=SMON	BM117210
CLND(2)=JOUT	BM117220
CLND(3)=SYR	BM117230
GOTO 300	BM117240
390 KMON=MOD(JSYR,12)	BM117250
IF(KMON.GT.0)GOTO 395	BM117260
KMON=12	BM117270
395 KYR=(JSYR-KMON)/12+YEAR	BM117280
CLND(1)=KMON	BM117290
CLND(2)=JOUT	BM117300
CLND(3)=KYR	BM117310
300 CONTINUE	BM117320
C	BM117330
CLND(3)=CLND(3)-1900	BM117340
C	BM117350
RETURN	BM117360
END	BM117370
	BM117380
	BM117390
	BM117400
SUBROUTINE CALA(CLND,WD)	BM117410
C *****	BM117420
C *	BM117430
C * THIS ROUTINE CONVERTS THE CALENDAR DATE INTO WORKING DAYS	BM117440
C *	BM117450
C *****	BM117460
INTEGER WD, YEAR, SMON, SDAY, SYR, WKDTB(120,31), CLND(3)	BM117470
COMMON/CCUN/SMON, SDAY, SYR, YEAR, WKDTB	BM117480
C	BM117490
CLND(2)=CLND(2)-1	BM117500
C	BM117510
ISYR=(SYR-YEAR)/12+SMON	BM117520
WD=C	BM117530
IYR=(CLND(3)-YEAR)/12+CLND(1)	BM117540
IF(IYR.LT.ISYR)GOTO 650	BM117550
IF(IYR.EQ.ISYR)GOTO 580	BM117560
DC 400 PI=ISYR,IYR	BM117570
IF(PI.EQ.ISYR)GOTO 410	BM117580
IF(PI.EQ.IYR)GOTO 420	BM117590
DC 440 PI=1,31	BM117600

	WD=WD+WKDTB(MI,MK)	BM117610
440	CONTINUE	BM117620
	GOTO 400	BM117630
420	LM=CLND(2)	BM117640
	DC 430 ML=1,LM	BM117650
	WD=WD+WKDTB(MI,ML)	BM117660
430	CONTINUE	BM117670
	GOTO 400	BM117680
410	DC 450 MJ=SDAY,31	BM117690
	WD=WD+WKDTB(MI,MJ)	BM117700
450	CONTINUE	BM117710
400	CONTINUE	BM117720
	GOTO 470	BM117730
580	LK=CLND(2)	BM117740
	IF(LK.LI.SDAY)GOTO 650	BM117750
	DC 460 LI=SDAY,LK	BM117760
	WD=WD+WKDTB(LI,LI)	BM117770
460	CONTINUE	BM117780
470	REP=WD	BM117790
	GOTO 500	BM117800
C		BM117810
650	PRINT *, ' ILLEGAL TARGET FINISH ** PLEASE CHECK '	BM117820
	REP = 0	BM117830
500	CONTINUE	BM117840
C		BM117850
	RETURN	BM117860
	END	BM117870

```

C*****
C
C          SSP          FORTRAN
C          -----
C          THIS PROGRAM COMPUTES THE STATISTICS AND PRINTS THE CUMULATIVE
C          PERCENTAGE FOR VARIOUS RESOURCES USAGE DATA PREPARED BY THE
C          SIMULATION PROGRAM.
C*****
C          INTEGER SHOPS(50),WCNTRS(200),TREF(50),MILSTN(50),START(3)
C          INTEGER DATA1(50,50,200)
C          CHARACTER*20 NAME,POL
C          COMMON /AAA/DATA1
C          COMMON/B/ICAL,IFIL,NRUN
C
C          REAL PARAMETERS :
C          READ(5,90) NAME,POL
C          90 FORMAT(A20,/,30X,A20)
C          READ(8,100)NCYCLS,NWC,NSHOPS,ILGW,IUP,NREPRT,NMILES
C          100 FORMAT(29X,15,////,29X,15,/,29X,15,/,29X,15,/,29X,15,
C          +      ///,29X,15,/,29X,15)
C
C          WRITE(6,*) ' '
C          WRITE(6,*) ' ENTER REPORTING OPTION CODE : '
C          WRITE(6,*) ' 1 = SHOPS ONLY '
C          WRITE(6,*) ' 2 = WORKCENTER ONLY '
C          WRITE(6,*) ' 3 = SHOPS & WORKCENTER '
C          READ(5,*) IPCL
C          IF(IPCL.EQ.1) NWC=0
C          IF(IPCL.EQ.2) NSHOPS=0
C
C          WRITE(6,*) ' ENTER CALENDAR OPTION CODE : '
C          WRITE(6,*) ' 0 = REPORT IN WORKING DATES '
C          WRITE(6,*) ' 1 = REPORT IN CALENDAR DATES '
C          READ(5,*) ICAL
C
C          IF(NREPRT.EQ.0) GO TO 104
C          SET FILE NUM. FOR OUTPUTS
C          NRUN=1
C          1 CONTINUE
C          IF(ICAL.EQ.0) IFIL=11
C          IF(ICAL.EQ.1) IFIL=15
C
C          IF(ICAL.EQ.1) THEN
C              READ(32,102) (TREF(I),I=1,NREPRT)
C              READ(1,99) (START(I),I=1,3)
C              ENDIF
C          IF(ICAL.EQ.1) READ(2,101) (TREF(I),I=1,NREPRT)
C          99 FORMAT(10X,3I5)
C          102 FORMAT(16)
C          101 FORMAT(20I4)
C
C          104 CONTINUE
C
C          IF(NWC.EQ.0) GO TO 1111

```

SSP00010
SSP00020
SSPC0030
SSPC0040
SSP00050
SSP00060
SSPC0070
SSPC0080
SSPC0090
SSPC0100
SSPC0110
SSPC0120
SSPC0130
SSPC0140
SSPC0150
SSP00160
SSPC0170
SSPC0180
SSPC0190
SSPC0200
SSPC0210
SSPC0220
SSPC0230
SSPC0240
SSPC0250
SSPC0260
SSPC0270
SSPC0280
SSP00290
SSPC0300
SSPC0310
SSPC0320
SSPC0330
SSPC0340
SSPC0350
SSPC0360
SSPC0370
SSPC0380
SSPC0390
SSP00400
SSPC0410
SSPC0420
SSPC0430
SSPC0440
SSPC0450
SSPC0460
SSPC0470
SSPC0480
SSPC0490
SSPC0500
SSPC0510
SSPC0520
SSPC0530
SSPC0540
SSPC0550

```

C      REPORT STAT OF RESOURCE USE BY WORK-CENTERS :
      IF(NRUN.EQ.1) THEN
        PRINT *, ' REPORTING WORK-CENTERS '
        WRITE(10,500) NAME,NCYCLS,PCL,ILOW,IUP
        WRITE(10,501) START
        ENDIF
      READ(13,105) (MCNTRS(I),I=1,NWC)
105  FORMAT (5X,I5)
      CALL ACCUM1(NCYCLS,NREPRT,NWC,25,MCNTRS,'WORK-CENT ',TREP,
1 'TIME ')
      IF(NSHOPS.EQ.0) GO TO 1114
C      REPORT STAT OF RESOURCE USE BY SHOP :
1111 IF(NRUN.EQ.1) THEN
      PRINT *, ' REPORTING SHOPS '
      WRITE(10,500) NAME,NCYCLS,PCL,ILOW,IUP
      WRITE(10,501) START
      ENDIF
      READ(4,101) (SHOPS(I),I=1,NSHOPS)
      CALL ACCUM1(NCYCLS,NREPRT,NSHOPS,26,SHOPS,'SHOP ',TREP,
1 'TIME ')
C      REPORT STAT OF MILE-STONES DATES :
1114 IF(NRUN.EQ.1) THEN
      PRINT *, ' REPORTING MILE-STONES '
      WRITE(10,500) NAME,NCYCLS,PCL,ILOW,IUP
      WRITE(10,501) START
      ENDIF
      READ(7,101) (MILSTN(I),I=1,NMILES)
      IF(ICAL.EQ.0) CALL ACCUM1(NCYCLS,1,NMILES,27,SHOPS,' ',
1 MILSTN,'MILE-STONE')
      IF(ICAL.EQ.1) CALL ACCUM1(NCYCLS,1,NMILES,57,SHOPS,' ',
1 MILSTN,'MILE-STONE')
C      REPORT STAT OF TOTAL MAN-HOURS PER SHOP :
      IF(NRUN.EQ.1) THEN
        PRINT *, ' REPORTING TOTAL MAN HOURS '
        WRITE(10,500) NAME,NCYCLS,PCL,ILOW,IUP
        WRITE(10,501) START
        ENDIF
        CALL ACCUM1(NCYCLS,1,NSHOPS,28,SHOPS,' ',SHOPS,
1 'SHOP ')
C
500  FORMAT(1H1,120(1H=),//,6X,'PROJECT NAME : ',A20,15,' RUN(S)',
+ ' USING ',A20,' INTENSITY RANGE ',15,' - ',15,/,120(1H=))
501  FORMAT(1H1,' PROJECT START DATE :',315,/)
      IF(NRUN.EQ.1) THEN
        PRINT *, ' CREATING OUTPUTS FOR GRAPHS '
        NFUN=2
        ICAL=ICAL+1
        IF(ICAL.EQ.1) ICAL=1
        IF(ICAL.EQ.2) ICAL=0
        REWIND 3
        REWIND 4

```

```

REWIND 7
REWIND 25
REWIND 26
REWIND 27
REWIND 28
GO TO 1
ENDIF

```

```

C
STOP
END

```

```

SUBROUTINE ACCUM1(NROW,NCOL1,NCOL2,IFILE,NAME1,HEAD1,NAME2,
1 HEAD2)

```

```

-----
CHARACTER*10 HEAD1,HEAD2
INTEGER DATA(50,50,200),ACCUM(10),NAME1(200),NAME2(200)
COMMON/AAA/DATA1
COMMON/B/ICAL,KFILE,NRUN

```

```

DO 20 I=1,NROW
DO 10 J=1,NCOL1
10 READ(IFILE,200,END=28) (DATA1(I,J,K),K=1,NCOL2)
200 FORMAT(10I8)
20 CONTINUE
26 CONTINUE

```

```

IF(IFILE.EQ.57) THEN

```

```

DO 41 I=1,NROW
DO 41 J=1,NCOL1
DO 41 K=1,NCOL2
ISAV=DATA1(I,J,K)/100
IYYY=MOD(DATA1(I,J,K),100)
DATA1(I,J,K)=ISAV+IYYY*10000

```

```

41 CONTINUE
ENDIF

```

```

IF(NRUN.EQ.2) GO TO 30
IF(HEAD2.EQ.'MILE-STONE') THEN
WRITE(10,114)
WRITE(10,136)
WRITE(10,140)
ENDIF

```

```

IF(HEAD2.EQ.'SHOP') THEN
WRITE(10,116)
WRITE(10,137)
WRITE(10,140)
ENDIF

```

```

30 CONTINUE

```

```

C
DO 22 ICOL=1,NCOL2
IF(NRUN.EQ.2) GO TO 32
IF (ICOL.EQ.1) THEN
IF(HEAD1.EQ.'WORK-CENT') WRITE(10,111)
IF(HEAD1.EQ.'SHOP') WRITE(10,113)
ELSE

```

```

SSP0111
SSP0112
SSP0113
SSP0114
SSP0115
SSP0116
SSP0117
SSP0118
SSP0119
SSP0120
SSP0121
SSP0122
SSP0123
SSP0124
SSP0125
SSP0126
SSP0127
SSP0128
SSP0129
SSP0130
SSP0131
SSP0132
SSP0133
SSP0134
SSP0135
SSP0136
SSP0137
SSP0138
SSP0139
SSP0140
SSP0141
SSP0142
SSP0143
SSP0144
SSP0145
SSP0146
SSP0147
SSP0148
SSP0149
SSP0150
SSP0151
SSP0152
SSP0153
SSP0154
SSP0155
SSP0156
SSP0157
SSP0158
SSP0159
SSP0160
SSP0161
SSP0162
SSP0163
SSP0164
SSP0165

```

```

      IF(HEAD1.EQ.'WORK-CENT ') WRITE(10,110)
      IF(HEAD1.EQ.'SHOP      ') WRITE(10,112)
      ENDIF
      IF(HEAD1.EQ.'WORK-CENT '.OR.HEAD1.EQ.'SHOP      ') THEN
        WRITE(10,132) HEAD1,NAME1(ICOL)
        WRITE(10,135)
        WRITE(10,140)
      ENDIF
32  CONTINUE

      DO 21 IREP=1,NCOL1
      CALL SORT(NROW,IREP,ICOL)
      K=1
      IF (NROW.GT.10) K=NROW/10 + .5
      DELN=NROW/10.
      ACCUM(1)=DATA1(K,IREP,ICOL)
      DO 19 I=2,10
      AN=I*DELN
      N=AN+.95
      IF (AN.GT.K) GOTO 25
      ACCUM(I)=ACCUM(I-1)
      GOTO 19
25  ACCUM(I)=DATA1(N,IREP,ICOL)
      K=K+1
19  CONTINUE
      AV=C.
      VAR=0.
      DO 192 I=1,NROW
      AV=AV+DATA1(I,IREP,ICOL)*1.
192  VAR=VAR+((DATA1(I,IREP,ICOL)*1.)**2)/(NROW*1.)
      AV=AV/NROW
      VAR=VAR-AV*AV
      IF(VAR.LT.C) VAR=0.
      STDEV=SQRT(VAR)
      ICIF=IREP
      IF(NCOL1.EQ.1) ICIR=ICOL

      IF(NROW.EQ.2) GO TO 33
      IF(ICAL.EQ.1) THEN
        WRITE(10,120) NAME2(ICIR),(ACCUM(I),I=1,10),AV,STDEV
      ELSE
        WRITE(10,124) NAME2(ICIR),(ACCUM(I),I=1,10),AV,STDEV
      ENDIF
33  CONTINUE
      IF(HEAD1.EQ.'WORK-CENT ')
      +  WRITE(MFILE,121)NAME1(ICOL),NAME2(ICIR),(ACCUM(I),I=1,10)
      IF(HEAD1.EQ.'SHOP      ')
      +  WRITE(MFILE+1,121)NAME1(ICOL),NAME2(ICIR),(ACCUM(I),I=1,10)
      IF(HEAD2.EQ.'MILE-STONE') THEN
      IF(MFILE.EQ.57) THEN
        DO 51 I=1,10
          IYYY=ACCUM(I)/10000
          ISAV=PCD(ACCUM(I),10000)
          ACCUM(I)=ISAV*100+IYYY
51  CONTINUE

```

SSPC160
SSPC161
SSPC166
SSPC169
SSPC170
SSPC171
SSPC172
SSPC173
SSPC174
SSPC175
SSPC176
SSPC177
SSPC178
SSPC179
SSPC180
SSPC181
SSPC182
SSPC183
SSPC184
SSPC185
SSPC186
SSPC187
SSPC188
SSPC189
SSPC190
SSPC191
SSPC192
SSPC193
SSPC194
SSPC195
SSPC196
SSPC197
SSPC198
SSPC199
SSPC200
SSPC201
SSPC202
SSPC203
SSPC204
SSPC205
SSPC206
SSPC207
SSPC208
SSPC209
SSPC210
SSPC211
SSPC212
SSPC213
SSPC214
SSPC215
SSPC216
SSPC217
SSPC218
SSPC219
SSPC220

```

      ENDIF
      WRITE(KFILE+2,122) NAME2(ICIF),(ACCLM(1),I=1,10)
      ENDIF
      IF(HEAD2.EQ.'SHOP' .AND.NRUN.EQ.1)
+     WRITE(14,122) NAME2(ICIF),(ACCU(1),I=1,10)
C
      21 CONTINUE
      22 CONTINUE
C
      RETURN
C
C     **** FORMATS ****
C
132 FORMAT(1H ,/,3X,A1C,1X,IS,/,3X,20('-',))
140 FORMAT(1H ,16X,'10%',4X,'20%',4X,'30%',4X,'40%',4X,'50%',
      1 4X,'60%',4X,'70%',4X,'80%',4X,'90%',3X,'100%',/,1X,120('-',),/)
120 FORMAT(1H ,6X,16,1C17,8X,F8.1,8X,F6.1)
124 FORMAT(1H ,6X,16,1C17,8X,F6.1,8X,F8.1)
121 FORMAT(15,1X,16,1X,10(17,1X))
122 FORMAT(15,8X,10(17,1X))
135 FORMAT(1H , 'PERIOD ENDED',24X,
+ 'C O N F I D E N C E   L E V E L',26X,'MEAN',9X,'STD DEV')
136 FORMAT(1H , 'MILESTONE',24X,
+ 'C O N F I D E N C E   L E V E L',26X,'MEAN',9X,'STD DEV')
137 FORMAT(1H , 'SHCPS',24X,
+ 'C O N F I D E N C E   L E V E L',26X,'MEAN',9X,'STD DEV')
110 FORMAT(1H1,5X,'RESOURCE USE (AVE. MAN-HOURS PER DAY) BY ',
+ 'WORK CENTER',/,1X,120(1H-))
111 FORMAT(1H ,/,5X,'RESOURCE USE (AVE. MAN-HOURS PER DAY) BY ',
+ 'WORK CENTER',/,1X,120(1H-))
112 FORMAT(1H1,/,5X,'RESOURCE USE (AVE. MAN-HOURS PER DAY) BY SHCP',
+ '/',1X,120(1H-))
113 FORMAT(1H ,/,5X,'RESOURCE USE (AVE. MAN-HOURS PER DAY) BY SHCP',
+ '/',1X,120(1H-))
114 FORMAT(1H ,5X,'DATES OF MILE-STONES ',/,1X,50('-',))
116 FORMAT(1H ,5X,'TOTAL MAN-HOURS BY SHCP',/,1X,35(1H-))
      END

      SUBROUTINE SORT(KLIMIT,ICCL1,ICCL2)

      INTEGER B(450),DATA1(50,50,200),VECCUT(450)
      COMMON/AAA/DATA1

      DO 10 I=1,KLIMIT
      B(I)=DATA1(I,ICCL1,ICCL2)
10  CONTINUE
      ALIMIT=KLIMIT*1.0
      ANUM=ALOG10(ALIMIT)/ALOG10(2.)
      NLM=ANUM
      IF(ANUM.GT.NLM) NLM=ANUM+1
      MM=2**NLM
      NRESID=MM-KLIMIT
      DO 101 L=1,NRESID
101  B(KLIMIT+L)=10000
      DO 6 I=1,NLM

```

SSP0221
SSP0222
SSP0223
SSP0224
SSP0225
SSP0226
SSP0227
SSP0228
SSP0229
SSP0230
SSP0231
SSP0232
SSP0233
SSP0234
SSP0235
SSP0236
SSP0237
SSP0238
SSP0239
SSP0240
SSP0241
SSP0242
SSP0243
SSP0244
SSP0245
SSP0246
SSP0247
SSP0248
SSP0249
SSP0250
SSP0251
SSP0252
SSP0253
SSP0254
SSP0255
SSP0256
SSP0257
SSP0258
SSP0259
SSP0260
SSP0261
SSP0262
SSP0263
SSP0264
SSP0265
SSP0266
SSP0267
SSP0268
SSP0269
SSP0270
SSP0271
SSP0272
SSP0273
SSP0274
SSP0275

K=2**1	SSP0276
M=MM/K	SSP0277
DC 4 J1=1,M	SSP0278
LSTART=K*(J1-1)+1	SSPC279
LENC=LSTART+K-1	SSP0280
K2=K/2	SSP0281
K1=C	SSPC282
J2=LSTART	SSPC283
1 IF(E(J2).LE.B(J2+K2)) GO TO 2	SSP0284
VECCUT(LSTART+K1)=B(J2+K2)	SSP0285
B(J2+K2)=999999	SSPC286
K2=K2+1	SSP0287
GO TO 3	SSP0288
2 VECCUT(LSTART+K1)=B(J2)	SSP0289
B(J2)=999999	SSPC290
J2=J2+1	SSPC291
K2=K2-1	SSPC292
3 J3=J2-LSTART+1	SSPC293
IF(J3.GT.(K/2)) GO TO 31	SSPC294
IF((J3+K2).GT.K) GO TO 31	SSPC295
K1=K1+1	SSP0296
GO TO 1	SSP0297
21 DC 32 L=LSTART,LEND	SSP0298
IF(E(L).GE.999999) GO TO 32	SSP0299
K1=K1+1	SSP0300
VECCUT(LSTART+K1)=B(L)	SSP0301
32 CONTINUE	SSP0302
4 CONTINUE	SSP0303
DC 5 J=1,KLIMIT	SSP0304
5 B(J)=VECCUT(J)	SSP0305
6 CONTINUE	SSP0306
DC 7 I=1,KLIMIT	SSP0307
DATA1(I,ICOL1,ICOL2)=B(I)	SSP0308
7 CONTINUE	SSP0309
RETRN	SSP0310
END	SSP0311

```

C*****
C
C          GRAPH      FORTRAN
C          -----
C    GRAPH READS THE OUTPUTS FROM SSP (OUTREP), AND MAKES
C    1. MILESTONE GRAPH
C    2. LOAD PROFILE
C    3. TOTAL MANHOUR GRAPH
C*****
C          CHARACTER*1 NL
C
C    READ OPTIONS
C
C    WRITE(6,*) ' *****'
C    WRITE(6,*) ' *      G R A P H S      *'
C    WRITE(6,*) ' *****'
C 1  WRITE(6,*) ' '
C    WRITE(6,*) ' SELECT GRAPH TYPE YOU WANT : '
C    WRITE(6,*) ' 1 = MILESTONE '
C    WRITE(6,*) ' 2 = LOAD PROFILE '
C    WRITE(6,*) ' 3 = TOTAL RESOURCE USAGE '
C    WRITE(6,*) ' 4 = TERMINATE EXECUTION '
C
C    READ(5,*) ITYPE
C    IF(ITYPE.LT.1.OR.ITYPE.GT.4) THEN
C      WRITE(6,*) ' INCORRECT TYPE.. PLEASE ENTER AGAIN '
C      GO TO 1
C    ENDIF
C
C    IF(ITYPE.EQ.1) CALL MILES
C    IF(ITYPE.EQ.2) CALL LOADS
C    IF(ITYPE.EQ.3) CALL TOTALS
C    IF(ITYPE.EQ.4) STOP
C
C    WRITE(6,*) ' '
C    WRITE(6,*) ' NEED MORE? Y IF YES, N IF NO. '
C    READ(5,100) NL
C 100 FORMAT(A1)
C*****
C***** BEFORE GO BACK REWIND ALL FILES
C    IF(NL.EQ.'Y') GO TO 1
C    STOP
C    END
C
C    SUBROUTINE MILES
C
C    ROUTINE FOR MILESTONE GRAPH
C
C    DIMENSION MILD(10,2),MID(10),MICO(10)
C
C    WRITE(6,*) ' ***** MILESTONE GRAPH ***** '
C 1  WRITE(6,*) ' ENTER MILESTONE NODE YOU WANT '
C    READ(5,*) MIL

```

GRAC0010
 GRAC0020
 GRAC0030
 GRAC0040
 GRAC0050
 GRAC0060
 GRAC0070
 GRAC0080
 GRAC0090
 GRAC0100
 GRAC0110
 GRAC0120
 GRAC0130
 GRAC0140
 GRAC0150
 GRAC0160
 GRAC0170
 GRAC0180
 GRAC0190
 GRAC0200
 GRAC0210
 GRAC0220
 GRAC0230
 GRAC0240
 GRAC0250
 GRAC0260
 GRAC0270
 GRAC0280
 GRAC0290
 GRAC0300
 GRAC0310
 GRAC0320
 GRAC0330
 GRAC0340
 GRAC0350
 GRAC0360
 GRAC0370
 GRAC0380
 GRAC0390
 GRAC0400
 GRAC0410
 GRAC0420
 GRAC0430
 GRAC0440
 GRAC0450
 GRAC0460
 GRAC0470
 GRAC0480
 GRAC0490
 GRAC0500
 GRAC0510
 GRAC0520
 GRAC0530
 GRAC0540
 GRAC0550

DC 3 I=1,1000	GRAC056
READ(1,*,END=5) MI,(MID(J),J=1,10)	GRAC057
IF(MI.EQ.MIL) GO TO 7	GRA0058
3 CONTINUE	GRA0059
	GRAC060
	GRA0061
5 WRITE(6,*) ' ** MILESTONE NODE NOT FOUND. PLEASE CHECK...'	GRAC062
REWIND 1	GRAC063
GO TO 1	GRA0064
C	GRAC065
7 II=1	GRAC066
DC 5 I=1,II	GRA0067
READ(2,*) MII,(MICO(J),J=1,10)	GRAC068
9 CONTINUE	GRAC069
IF(MI.NE.MII) THEN	GRAC070
WRITE(6,*) ' ** MISMATCHED MILESTONE ... CHECK. '	GRA0071
STOP	GRA0072
ENDIF	GRAC073
C	GRA0074
DC 11 I=1,10	GRAC075
MILD(I,1)=MICO(I)	GRAC076
MILD(I,2)=MID(I)	GRA0077
11 CONTINUE	GRA0078
C	GRA0079
C	GRA0080
C	GRAC081
CALL GRAPH ROUTINE	GRAC082
CALL MILE(MI,MILD)	GRAC083
C	GRAC084
REWIND 1	GRA0085
REWIND 2	GRAC086
RETURN	GRAC087
END	GRAC088
	GRA0089
SUBROUTINE MILE(NUM,DATES)	GRAC090
C	GRA0091
MILESTONE GRAPH	GRAC092
	GRA0093
INTEGER NUM,DATES(10,2),DAT(10)	GRA0094
DIMENSION CUN(10),WORK(10)	GRAC095
	GRAC096
CALL TK4015(120,0)	GRAC097
C	GRA0098
CALL PRTPLOT(79,6)	GRAC099
IF(IRETR.NE.0) STOP	GRAC100
CALL PAGE(15.,11.)	GRAC101
CALL XNAME('WORKING DAYS\$',12)	GRAC102
CALL YNAME('CONFIDENCE LEVEL(%)\$',100)	GRAC103
CALL AREA2D(13.,8.)	GRAC104
CALL HEADIN('SIMULATED REALIZATION\$',21,1.7,2)	GRAC105
CALL HEADIN('OF MILESTONE #\$',14,1.7,2)	GRA0106
CALL HEIGHT(.2)	GRAC107
CALL INTNO(NUM,8.5,8.45)	GRAC108
CALL RESET('HEIGHT')	GRA0109
DO 9 I=1,10	GRAC110
INTER=(DATES(10,2)-DATES(1,2))/10*41	

```

      IF (INTER.LT.1) THEN
        IF (I.LE.2) INTER=1
        IF (I.GT.2) INTER=10*(I-2)
        GO TO 8
      ENDF
9  CONTINUE
8  CALL GRAF(DATES(1,2),INTER,DATES(10,2)+INTER,C.,10.,100.)
   DC 10 I=1,10
      CCN(I)=I*10
      WCRK(I)=DATES(1,2)
      DAT(I)=DATES(I,1)
10  CONTINUE
   CALL MARKER(15)
   CALL CURVE (WCRK,CCN,10,1)
   DC 20 I=1,10
      CALL RLINT(DAT(I),WCRK(I),CCN(I))
20  CONTINUE
   CALL ENDFL(0)
   CALL DONEPL
   RETURN
   END

SUBROUTINE LOADS
C
C  ROUTINE FOR LOAD PROFILE
C
   INTEGER ARRAY(100,4),LLCAD(10),CONFID,PASS(100,3)

   WRITE(6,*) ' ***** L O A D   P R O F I L E   ***** '
   WRITE(6,*) ' CHOOSE OPTION : 0 = WORKCENTER LEVEL '
   WRITE(6,*) '                   1 = SHOP           LEVEL '
   READ(5,*) ICPT
   WRITE(6,*) ' CHOOSE OPTION : 0 = IN WORKING DAYS '
   WRITE(6,*) '                   1 = IN CALENDAR DATE '
   READ(5,*) ICAL
C  READ NUMBER OF REPORTING DATES
   READ(12,100) NREPT
100  FORMAT(12(/),29X,15)
C
C  WORKCENTER LOAD GRAPH
C  -----
1  IF (ICPT.EQ.0) THEN
      WRITE(6,*) ' ENTER WORKCENTER AND CONFIDENCE LEVEL (IN %) '
      READ(5,*) IWC,CONFID

      KIL=0
      ITN=0
      DO 3 I=1,1000
        READ(3,*,END=5) IWC1,IDAT1,(LLCAD(I),J=1,10)
        READ(4,*,END=5) IWC2,IDAT2,(LLCAD(I),J=1,10)
        IF (IWC1.NE.IWC2) THEN
          WRITE(6,*) ' ERROR IN FILE 3 OR 4..CHECK '
          STOP
        ENDF
      3  CONTINUE

```

GRA0111
 GRA0112
 GRA0113
 GRA0114
 GRA0115
 GRA0116
 GRA0117
 GRA0118
 GRA0119
 GRA0120
 GRA0121
 GRA0122
 GRA0123
 GRA0124
 GRA0125
 GRA0126
 GRA0127
 GRA0128
 GRA0129
 GRA0130
 GRA0131
 GRA0132
 GRA0133
 GRA0134
 GRA0135
 GRA0136
 GRA0137
 GRA0138
 GRA0139
 GRA0140
 GRA0141
 GRA0142
 GRA0143
 GRA0144
 GRA0145
 GRA0146
 GRA0147
 GRA0148
 GRA0149
 GRA0150
 GRA0151
 GRA0152
 GRA0153
 GRA0154
 GRA0155
 GRA0156
 GRA0157
 GRA0158
 GRA0159
 GRA0160
 GRA0161
 GRA0162
 GRA0163
 GRA0164
 GRA0165

```

      IF(IWC1.NE.IWC .AND.KIL.EQ.1) GO TO 7
      IF(IWC1.EQ.IWC ) THEN
        ITN=ITN+1
        ARRAY(ITN,1)=IDAT1
        ARRAY(ITN,2)=IDAT2
        ARRAY(ITN,4)=LLOAD(CONFID/10)
        KIL=1
        ENDIF
3     CONTINUE

5     IF(KIL.EQ.1) GO TO 7
      WRITE(6,*) ' ** WORKCENTER NOT FOUND. PLEASE CHECK...'
      REWIND 3
      REWIND 4
      GO TO 1

C
7     CONTINUE
C     SHOP CAPACITY (THERE IS NO WORKCENTER-LEVEL CAP.)
      KILL=0
      ITNN=0
      DO 9 I=1,1000
        ISW=0
        READ(10,*,END=11) IWCT, IDATE, CAPA
        ISHP=IWC/100
        IF(IWCT.NE.ISHP .AND.KILL.EQ.1) GO TO 13
14    CONTINUE
        IF(IWCT.EQ.ISHP ) THEN
          ITNN=ITNN+1
          IF(ARRAY(ITNN,1).GT.IDATE) THEN
            ARRAY(ITNN,3)=CAPA+.5
            IF(ISW.EQ.0) THEN
              ISAVEN=ITNN
              ISW=1
            ENDIF
            GO TO 14
          ENDIF
          IF(ITNN.GT.NREPT) GO TO 8
          IF(ARRAY(ITNN,1).LE.IDATE.AND.ISW.EQ.0) GO TO 14
          IF(ARRAY(ITNN,1).LE.IDATE.AND.ISW.EQ.1) ITNN=ISAVEN
          KILL=1
        ENDIF
9     CONTINUE

11    IF(KILL.EQ.1) GO TO 13
      WRITE(6,*) ' ** SHOP NOT FOUND IN CAP. PLEASE CHECK...'
      GO TO 99
13    CONTINUE

C
      IF(ICAL.EQ.0) IK=1
      IF(ICAL.EQ.1) IK=2
      DO 15 I=1,NFEPT
        PASS(I,1)=ARRAY(I,IK)
        PASS(I,2)=ARRAY(I,3)
        PASS(I,3)=ARRAY(I,4)
15   CONTINUE

```

```

GRAC1660
GRAC1670
GRAC1680
GRAC1690
GRAC1700
GRAC1710
GRAC1720
GRAC1730
GRA01740
GRA01750
GRAC1760
GRA01770
GRAC1780
GRAC1790
GRAC1800
GRAC1810
GRAC1820
GRA01830
GRA01840
GRA01850
GRAC1860
GRAC1870
GRA01880
GRAC1890
GRA01900
GRAC1910
GRAC1920
GRAC1930
GRAC1940
GRA01950
GRA01960
GRA01970
GRA01980
GRA01990
GRA02000
GRAC2010
GRAC2020
GRAC2030
GRAC2040
GRA02050
GRA02060
GRAC2070
GRAC2080
GRAC2090
GRAC2100
GRAC2110
GRA02120
GRA02130
GRAC2140
GRAC2150
GRAC2160
GRAC2170
GRAC2180
GRAC2190
GRAC2200

```

C		GRAC221
C	CALL GRAPH ROUTINE	GRAC222
C		GRAC223
C	CALL LOAD(INC ,CONFID,ICAL,NREPT,PASS)	GRAC224
C		GRAC225
99	REWIND 3	GRAC226
	REWIND 4	GRAC227
	REWIND 10	GRAC228
	REWIND 12	GRAC229
	RETURN	GRAC230
	ENDIF	GRAC231
C		GRAC232
C	SHOP LOAD GRAPH	GRAC233
C	-----	GRAC234
21	IF(IOPT.EQ.1) THEN	GRAC235
	WRITE(6,*) ' ENTER SHOP NUMEER AND CONFIDENCE LEVEL (IN %)'	GRAC236
	READ(5,*) ISHOP,CONFID	GRAC237
		GRAC238
	KIL=0	GRAC239
	ITN=0	GRAC240
	DO 23 I=1,1000	GRAC241
	READ(7,*,END=25) ISH1,IDAT1,(LLDAD(J),J=1,10)	GRAC242
	READ(8,*,END=25) ISH2,IDAT2,(LLDAD(J),J=1,10)	GRAC243
	IF(ISH1.NE.ISH2) THEN	GRAC244
	WRITE(6,*) ' ERROR IN FILE 7 OR 8..CHECK'	GRAC245
	STOP	GRAC246
	ENDIF	GRAC247
	IF(ISH1.NE.ISHOP.AND.KIL.EQ.1) GO TO 27	GRAC248
	IF(ISH1.EQ.ISHOP) THEN	GRAC249
	ITN=ITN+1	GRAC250
	ARRAY(ITN,1)=IDAT1	GRAC251
	ARRAY(ITN,2)=IDAT2	GRAC252
	ARRAY(ITN,4)=LLDAD(CONFID/10)	GRAC253
	KIL=1	GRAC254
	ENDIF	GRAC255
23	CONTINUE	GRAC256
		GRAC257
25	CONTINUE	GRAC258
	IF(KIL.EQ.1) GO TO 27	GRAC259
	WRITE(6,*) ' ** SHOP NUMBER NOT FOUND. PLEASE CHECK...'	GRAC260
	REWIND 7	GRAC261
	REWIND 8	GRAC262
	GO TO 21	GRAC263
C		GRAC264
27	CONTINUE	GRAC265
C	CAPACITY	GRAC266
	KILL=0	GRAC267
	ITNN=0	GRAC268
	DO 29 I=1,1000	GRAC269
	ISM=0	GRAC270
	READ(10,*,END=31) ISHP,IDATE,CAFA	GRAC271
	IF(ISHP.NE.ISHOP.AND.KILL.EQ.1) GO TO 33	GRAC272
34	CONTINUE	GRAC273
	IF(ISHP.EQ.ISHOP) THEN	GRAC274
	ITNN=ITNN+1	GRAC275

IF(ARRAY(ITNN,1).GT.IDATE) THEN	GRA0276C
ARRAY(ITNN,3)=CAPA+.5	GRA0277C
IF(ISW.EQ.0) THEN	GRA0278C
ISAVEN=ITNN	GRA0279C
ISW=1	GRA0280C
ENDIF	GRA0281C
GO TO 34	GRA0282C
ENDIF	GRA0283C
IF(ITNN.GT.NREPT) GO TO 28	GRA0284C
IF(ARRAY(ITNN,1).LE.IDATE.AND.ISW.EQ.0) GO TO 34	GRA0285C
28 IF(ARRAY(ITNN,1).LE.IDATE.AND.ISW.EQ.1) ITNN=ISAVEN	GRA0286C
KILL=1	GRA0287C
ENDIF	GRA0288C
29 CONTINUE	GRA0289C
	GRA0290C
31 IF(KILL.EQ.1) GO TO 33	GRA0291C
WRITE(6,*) ' ** SHOP NUMEER NOT FOUND IN CAP. PLEASE CHECK...'	GRA0292C
GO TO 999	GRA0293C
33 CONTINUE	GRA0294C
C	GRA0295C
IF(ICAL.EQ.C) IK=1	GRA0296C
IF(ICAL.EQ.1) IK=2	GRA0297C
DO 35 I=1,NREPT	GRA0298C
PASS(I,1)=ARRAY(I,IK)	GRA0299C
PASS(I,2)=ARRAY(I,3)	GRA0300C
PASS(I,3)=ARRAY(I,4)	GRA0301C
35 CONTINUE	GRA0302C
C	GRA0303C
C	GRA0304C
C	GRA0305C
CALL LOAD(ISHOP,CONFID,ICAL,NREPT,PASS)	GRA0306C
C	GRA0307C
999 REWIND 7	GRA0308C
REWIND 8	GRA0309C
REWIND 10	GRA0310C
REWIND 12	GRA0311C
RETURN	GRA0312C
ENDIF	GRA0313C
	GRA0314C
END	GRA0315C
	GRA0316C
	GRA0317C
SUBROUTINE LOAD(SHOPNU,LEVEL,CAL,NUM,ARRAY)	GRA0318C
	GRA0319C
	GRA0320C
INTEGER CAL, LEVEL, ARRAY(100,3),SHOPNU	GRA0321C
INTEGER DAT(100)	GRA0322C
DIMENSION CAP(100),AVE(100),COUNT(100)	GRA0323C
	GRA0324C
C	GRA0325C
CALL PRTPRT(79,6)	GRA0326C
CALL TK4015(120,0)	GRA0327C
IF(IRETR.NE.0) STOP	GRA0328C
CALL PAGE(15.,11.)	GRA0329C
CALL YNAME('AVE. HOURS OF LOADS',100)	GRA0330C
CALL XNAME('DATE\$',4)	

CALL AREA2D (13.,8.)	GRA0331
CALL ANCNUM	GRA0332
CALL HEADIN('SIMULATED LOAD PROFILE\$',22 ,1.7,1)	GRA0333
CALL HEIGHT(.2)	GRA0334
CALL MESSAG('FOR RESOURCE: \$',14 ,5.,8.)	GRA0335
CALL MESSAG('CONFIDENCE LEVEL:\$',17,5.,8.4)	GRA0336
CALL INTNO(SHCPNU ,8.5 ,8.)	GRA0337
CALL INTNO(LEVEL,8.5,8.4)	GRA0338
CALL RESET('HEIGHT')	GRA0339
DC 10 I=1,NUM	GRA0340
DAT(I)=ARRAY(I,1)	GRA0341
CAP(I)=ARRAY(I,2)	GRA0342
AVE(I)=ARRAY(I,3)	GRA0343
COUNT(I)=1	GRA0344
10 CONTINUE	GRA0345
NUM1=-NUM	GRA0346
INT1=0	GRA0347
DD 5 I=1,10	GRA0348
INTT=CAP(I)/10	GRA0349
IF (INTT.GT.INT1) THEN	GRA0350
INT1=INTT	GRA0351
ICAP=CAP(I)	GRA0352
ENDIF	GRA0353
5 CONTINUE	GRA0354
DC 9 I=1,10	GRA0355
INTER=DAT(NUM)/10**I	GRA0356
IF (INTER.LT.1) THEN	GRA0357
INTER=10**((I-2)	GRA0358
GO TO 8	GRA0359
ENDIF	GRA0360
9 CONTINUE	GRA0361
8 CALL GRAF(.5,1.,COUNT(NUM)+.5,0.,INT1,ICAP+INT1)	GRA0362
CALL HEIGHT(.1)	GRA0363
DC 11 I=1,NUM	GRA0364
CALL INTNO(DAT(I) ,(I/1.7+.05*I),-.2)	GRA0365
11 CONTINUE	GRA0366
CALL RESET('HEIGHT')	GRA0367
CALL RESET('BARDC')	GRA0368
C CALL V3AFS(COUNT,'BASE',CAP,NUM)	GRA0369
C CALL V3AFS(COUNT,'BASE',AVE,NUM)	GRA0370
C CALL BARSHD(COUNT,0,CAP,NUM,.1,90.,0,C,C,0)	GRA0371
C CALL BARSHD(COUNT,0,AVE,NUM,.1,90.,0,C,C,0)	GRA0372
CALL STEP	GRA0373
CALL CURVE(COUNT,AVE,NUM,0)	GRA0374
CALL STEP	GRA0375
CALL DASH	GRA0376
CALL CURVE(COUNT,CAP,NUM,0)	GRA0377
CALL ENDCPL(0)	GRA0378
CALL DONEPL	GRA0379
RETURN	GRA0380
END	GRA0381
	GRA0382
	GRA0383
	GRA0384
	GRA0385

SUBFOLIOE TCTALS

C	ROUTINE FOR TOTALS	GRA0386C
C	INTEGER TOT(10),ITOT(10)	GRA0387C
	WRITE(6,*) * ***** TOTAL RESOURCE USAGE *****	GRA0388C
1	WRITE(6,*) * ENTER SHOP NUMBER , OR 9999 FOR GRAND TOTAL *	GRA0389C
	READ(5,*) ISHOP	GRA0390C
	IF(ISHOP.EQ.9999) GO TO 50	GRA0391C
	DO 3 I=1,1000	GRA0392C
	READ(9,*,END=5) ISHP,(TOT(J),J=1,10)	GRA0393C
	IF(ISHP.EQ.ISHOP) GO TO 7	GRA0394C
3	CONTINUE	GRA0395C
		GRA0396C
		GRA0397C
		GRA0398C
5	WRITE(6,*) * ** SHOP NUMBER NOT FOUND. PLEASE CHECK... *	GRA0399C
	REWIND 9	GRA0400C
	GO TO 1	GRA0401C
C		GRA0402C
	7 CONTINUE	GRA0403C
C		GRA0404C
C	CALL GRAPH ROUTINE	GRA0405C
C		GRA0406C
	CALL TOTAL(ISHOP,TOT)	GRA0407C
	REWIND 9	GRA0408C
	RETURN	GRA0409C
C		GRA0410C
	50 CONTINUE	GRA0411C
	DO 12 I=1,10	GRA0412C
12	ITOT(I)=0.	GRA0413C
	DO 13 I=1,1000	GRA0414C
	READ(9,*,END=15) ISHP,(TOT(J),J=1,10)	GRA0415C
	DO 14 K=1,10	GRA0416C
14	ITOT(K)=ITOT(K)+TOT(K)	GRA0417C
13	CONTINUE	GRA0418C
C		GRA0419C
	15 CONTINUE	GRA0420C
C		GRA0421C
C	CALL GRAPH ROUTINE	GRA0422C
C		GRA0423C
	CALL TOTAL(ISHOP,ITOT)	GRA0424C
	REWIND 9	GRA0425C
	RETURN	GRA0426C
	END	GRA0427C
		GRA0428C
		GRA0429C
		GRA0430C
	SUBROUTINE TOTAL(SHOPN,TOT)	GRA0431C
		GRA0432C
C	TOTAL GRAPH. IF SHOPN=9999, THEN TOT IS GRAND TOTAL	GRA0433C
		GRA0434C
	DIMENSION CON(10),TTT(10)	GRA0435C
	INTEGER SHOPN,TOT(10)	GRA0436C
C	CHARACTER*5 SHOPN	GRA0437C
C	REAL TOT(10)	GRA0438C
	CALL TK4C15(120,0)	GRA0439C
		GRA0440C

IF(I RETR.NE.0) STOP	GRA04410
CALL PAGE(15.,11.)	GRA04420
CALL XNAME('MAN HOURS\$',100)	GRA04430
CALL YNAME ('CONFIDENCE LEVEL(%)\$',100)	GRA04440
CALL AREA2D (13.,8.)	GRA04450
CALL HEADIN('SIMULATED TOTAL REQUIREMENTS\$',27,1.7,2)	GRA04460
CALL HEADIN('FOR RESOURCE #\$',14,1.7,2)	GRA04470
CALL HEIGHT(.2)	GRA04480
C ***** CHECK	GRA04490
IF(SHCPN.NE.9999) CALL INTNO(SHCPN,8.5,8.45)	GRA04500
IF(SHCPN.EQ.9999) CALL MESSAG('ALL\$',3,8.5,8.45)	GRA04510
CALL RESET('HEIGHT')	GRA04520
DC 9 I=1,10	GRA04530
INTER=TCT(10)/10**I	GRA04540
IF(INTER.LT.1) THEN	GRA04550
INTER=10** (I-2)	GRA04560
GO TO 8	GRA04570
ENDIF	GRA04580
9 CONTINUE	GRA04590
8 CALL GRAF(TCT(1),INTER,TOT(10)+INTER,C.,10.,100.)	GRA04600
DC 10 I=1,10	GRA04610
CGN(I)=I*10.	GRA04620
TTT(I)=TCT(I)	GRA04630
10 CONTINUE	GRA04640
CALL PARKER(15)	GRA04650
CALL CURVE (TTT,CJN,10,1)	GRA04660
DC 20 I=1,10	GRA04670
CALL RLINT(TOT(I),TTT(I),CGN(I))	GRA04680
20 CONTINUE	GRA04690
CALL ENDPL(0)	GRA04700
CALL DONEPL	GRA04710
RETURN	GRA04720
END	GRA04730

END
DITIC

7-86